Anomaly detection in a Mobile Data Network

Jason Paul Salzwedel

University of Cape Town

Data Science Minor Dissertation

STA5079W

Anomaly detection in a Mobile Data Network

Author:
Jason Salzwedel

Supervisor:
Mzabalazo Ngwenya

April 4, 2019

Contents

Chapter 1

Introduction

This project investigated and tested anomaly detection in a mobile network operating in South Africa. This introduction starts with an overview of anomaly detection followed by a brief description of the mobile network with a focus on the network elements that this project focused on. An overview of how anomaly detection has been historically done in the network along with its strengths and weaknesses is given. This is followed by a overview of some non-model based approaches that has been investigated in the industry.

1.1 Anomaly Detection

Anomaly detection or outlier detection is the process of identifying observations within a data set that differ substantially from the majority of the other observations. An outlier is defined by [1] as "An outlying observation, or outlier, is one that appears to deviate markedly from other members of the sample in which it occurs". This definition is taken a step further in [2] with the following definition "an outlier is an observation that deviates so much from other observations as to arouse suspicions that it was generated by a different mechanism". It is this "different mechanism" which the outlier points to which is ultimately of interest. This "different mechanism" in the context of this project is potentially a network fault or network performance issue which needs to be addressed to prevent a degraded service. An alternative reason for the the presence of the outlier is error. This could be be due to any number of errors in the process of measuring, collecting, processing or storing the data.
There is a vast amount of literature covering anomaly detection available. An overview is given in a survey of anomaly detection in [3]. This discusses different anomaly detection approaches taken in different environments. More recently, [4] focuses on comparing different unsupervised anomaly detection algorithms on various publicly available multivariate datasets. Both [3] and [4] discuss three topics in their introductions. These include the type of anomaly, the data labels available and the output of the anomaly detection technique. Five different categories of anomalies are covered. These are local, global, point, collective and contextual. An illustration of a local and a global anomaly are shown in Figure 1.1. Point x is far from any other groups of data points making it a global outlier. Point o is close to the cluster on the right of the diagram, but is far enough away from the local cluster to be able to identify it as a local outlier. Both the local and global anomalies are examples of point anomalies as they are separable from the other observations through their distances from the other observations. Collective anomalies are represented by a set of observations that are not normally observed together. Individually these observation are not considered anomalous, but as they are not normally observed together, an anomalous event is indicated when they are. Contextual anomalies are

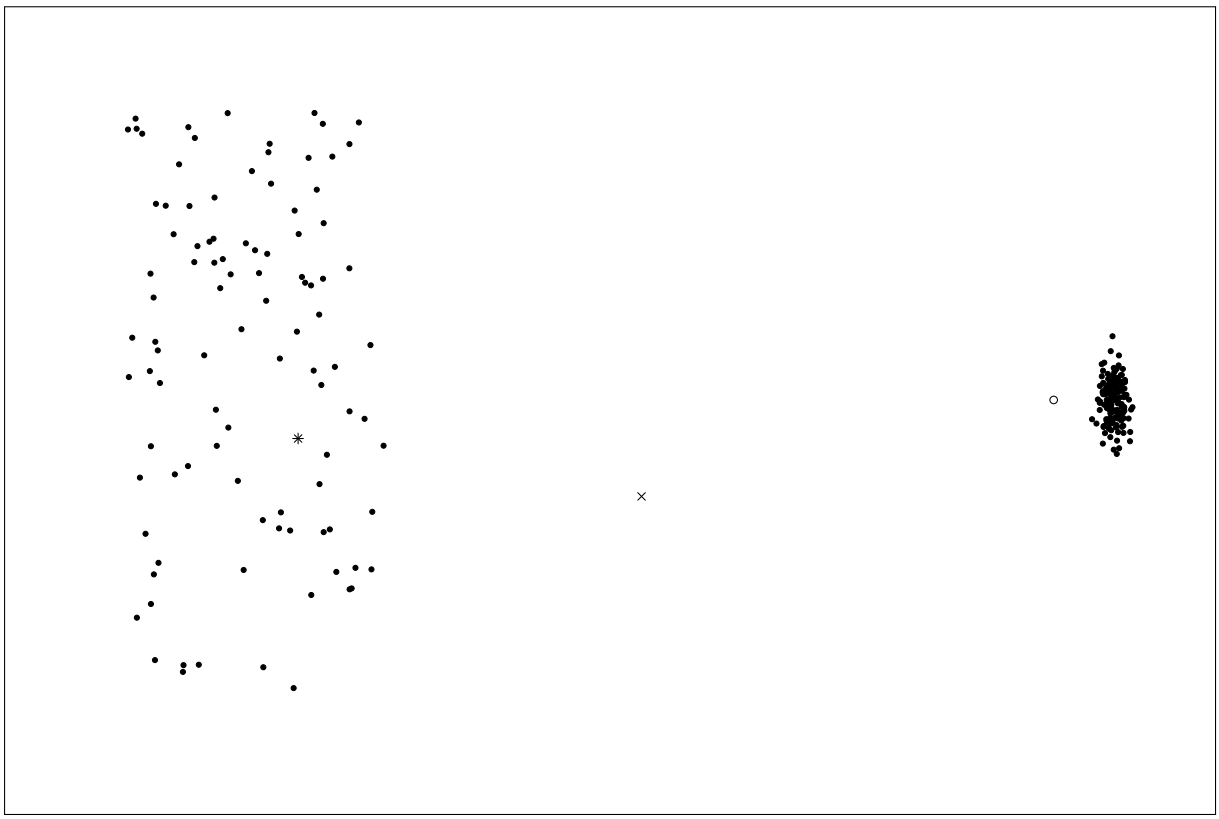


Figure 1.1: Example of a Global (x) and Local (o) outlier.

examples of data points that fall within the expected range, but not for that particular type of observation. For example, the outside temperature ranges between 4 and 40 degrees Celsius. An observation with a value of 40 degrees may not be an outlier if it was observed in Summer time, but if it was observed in Winter time, it would be an anomaly. Within the context of time the observation is an anomaly. The point marked as a * in the middle of the cluster on the left hand side of Figure 1.1 is an example of a contextual anomaly. This is in the way that all the observations are dots and the star differs in the context of shape. According to [4] most unsupervised anomaly detection algorithms cater for detecting point anomalies. It is possible to address the contextual and collective anomalies with these approaches through features engineering and the inclusion of the context as a feature of the data set.

Different approaches to anomaly detection can be taken dependent of the labels available in the dataset. If there are labels that indicate whether the observation is normal or anomalous, then supervised learning approaches can be used. As anomalies are by definition rare, supervised learning approaches would need to cater for unbalanced data sets. If the dataset contains only normal data points, semi-supervised approaches can be used. Here the model is trained to identify normal behavior and when presented with an anomalous observation a deviation from normal is detected. Lastly, if there are no labels available, then unsupervised anomaly detection methods are to be used. This last category is most prevalent due to the scarcity of labeled data sets. This study falls in this latter category as the data contains no labels.

The output given by the anomaly detection techniques can either be a label, indicating that a observation is normal or an outlier, or it can be a score, indicating how much of an outlier an observation is.

Unsupervised-learning approaches can be subdivided based on the underlying intrinsic characteristics of the dataset that are leveraged to identify outliers. Statistical based,

density based, distance based and deviation based approaches are identified in [5]. According to [6] statistical-based approaches assume the data comes from a specific distribution. Outliers are identified through a discordancy test, with different tests being applied depending on the availability of the distribution parameters, the predetermined number of outliers and the types of outliers expected. The majority of these discordancy tests only cater for a univariate datasets. A second statistics based approach referred to as a depth based approach, where observations are assigned a depth in a multidimensional space, with outliers tending to have smaller depths. Density based approaches including Local Outlier Factor [7] make use of the local densities of the observations. Observations that are in a lower density area than the surrounding neighborhood are identified as outliers. A score or, a factor is assigned to each observation. Observations assigned factors near to one are considered normal with outliers assigned values greater than one. The higher the value the more of an outlier it is considered to be. Distance based approaches make use of the relative distance between the observations. The first such approach was introduced in [6] where outliers were identified as observations that had the highest distances to the k^{th} nearest neighbor. Deviation based approaches base the outlier identification on the feature characteristics of the observations.

1.2 Mobile Network Environment

This study took place in the context of a mobile network operator operating in South Africa. The network architecture follows the 3GPP [8] specifications and provides voice and data services to over 40 million voice and data customers [9]. The network provides services through various radio access technologies, including UMTS, HSPA, HSPA+, LTE, as well as fixed network technologies including fiber to the home. An overview, as from [10], of a basic 3GPP Access PLMN supporting Circuit Switched and Packet Switched services is shown in Figure 1.2. This study focuses on the data consisting of two weeks worth of data generated by the SGW elements in the network. The functions of the SGW according to [10] are listed in Table 1.1.

Table 1.1: SGW Functions listing in Technical Specification (TS) 23.002, 3rd Generation Partnership Project (3GPP).

SGW Functions
The local Mobility Anchor point for inter-eNodeB handover
Mobility anchoring for inter-3GPP mobility
ECM-IDLE mode downlink packet buffering and initiation of network triggered SRPs
Lawful Interception
Packet routeing and forwarding
Transport level packet marking in the uplink and the downlink
Accounting on user and QCI granularity for inter-operator charging
Event reporting (change of RAT, etc.) to the PCRF
Uplink and downlink bearer binding towards 3GPP accesses
Uplink bearer binding verification with packet dropping of “misbehaving UL traffic”
Mobile Access Gateway (MAG) functions if PMIP-based S5 or S8 is used
Support necessary functions in order for enabling GTP/PMIP chaining functions.

Operations an maintenance of the network is provided using the FCAPS framework. FCAPS is an acronym for fault, Configuration, accounting/administration, performance and security. The current approach taken to monitor the network for faults or performance

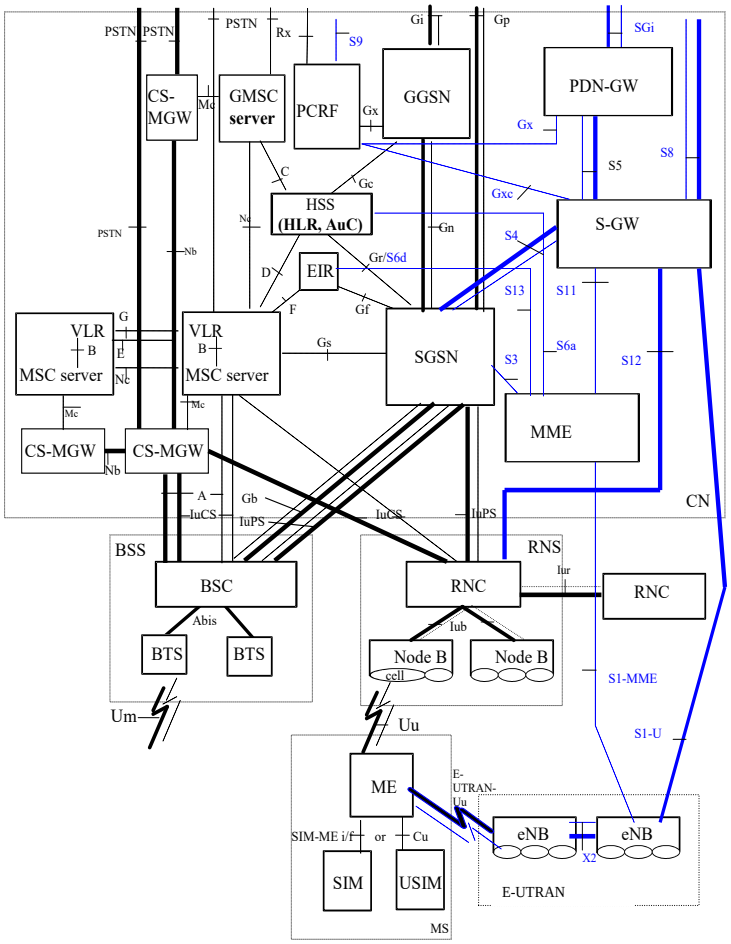


Figure 1.2: Basic Configuration of a 3GPP Access PLMN.

degradations start with the definition of key performance indicators (KPIs) based on domain knowledge. These KPIs can be grouped into 3 main categories [11], including success rate indicators such as the PDP activation success rate, failure rate indicators such as the call drop rate and neutral indicators such as the number of concurrently attached subscribers. This manual process of defining the KPIs starts with understanding of the network architecture, the element capacities, the redundancy strategy, the functions and procedures to be performed by the network elements, the signaling flows between the network elements required to perform these functions, the services provided and the associated traffic models as well as the customer life cycle. After the key performance indicators have been defined, the relevant counters required to report these KPI are identified within the data generated by the network elements. The next step is defining the thresholds for alarming on each KPI. The advantage of this approach is that it is tailored to the specific network configuration deployed and the equipment used. A disadvantage of this approach is that not all network functions are actively monitored, leaving blind spots. Added to this, the network is dynamic, the KPIs monitored as well as the associated thresholds need to be updated. Both these shortcomings could be addressed by increasing the size of the team that performs this task, but this is cost prohibitive. Table 1.2 [9] shows that the network is having to support more subscribers each year, while the average revenue per subscriber is decreasing each year. As stated in [11], the complexity of mobile networks is increasing, this together with the reducing ARPU drives the need to reducing the human workload needed for fault detection and performance management of the network. This increasing network complexity can be attributed in part to the following list of factors.

- Increasing number of subscribers using the network with an increase in the usage per subscriber
- Higher smart-phone penetration and associated application usage
- Higher penetration of Voice over LTE services
- Increasing number of services and complexity of services supported by the networks
- Increasing number of network element types with each successive network generation deployed and supported in parallel as well as the increase in complexity associated with the inter-working between each successive networking generation.
- Increasing number of network elements deployed in the network
- Increased demand for IoT support
- Complexity added with the deployment of NFV and SDN in parallel with traditional architectures.

This combination of increasing network complexity as well as the downward pressure on the costs associated with network management was the motivation for this study. It aims to address the first of the short comings of the model based approach currently used, namely to provide visibility on the blind spots on the network through anomaly detection in the high dimensional data generated by the network.

Table 1.2: ARPU and Subscriber base counts.

Year	ARPU	Active Subscribers	Data Subscribers	IOT subscribers
2011	R183	22 880 000		
2012	R157	28 941 000		
2013	R129	30 348 000		
2014	R125	31 520 000	15 172 000	1 443 000
2015	R113	32 115 000	16 595 000	1 760 000
2016	R112	34 178 000	18 704 000	2 264 000
2017	R111	37 131 000	19 549 000	2 979 000
2018	R101	41 635 000	20 347 000	3 628 000

1.3 Related Work

The following section focuses on anomaly detection in the field of communication networks, with an emphasis on mobile networks. The main aspects considered are; the network environment, the data source and associated data sets, what are considered to be anomalies, the goal to be achieved and lastly the detection techniques used.
An improved anomaly detection and diagnosis Framework for Mobile Network Operators is introduced in [11]. The study aims to expand on the authors previous framework created in [12] by automatically creating the normal profiles based on anomaly free data. This is achieved through a statistical primitive, derived from the two sample Kolmogorov-Smirnov test. An anomaly free data set obtained from 3G cell performance data, consisting of 12 performance indicators that were manually categorized as success indicator, failure or Neutral indicators. The study considered anomalies to be states that do not match a predefined normal profile. The main difference taken between [11] and this study, is that the data available in this cannot be considered to be anomaly free. Using the approach taken in [11] on this data set would result in any anomalous data to be considered as a normal in all future data sets.
An ensemble learning approach referred to as a "super learner" for detecting three predefined DNS anomalies in a European Mobile Network Operator (MNO) is proposed in [13]. The machine learning and production environment is based on Big-DAMA which supports the training of different models in parallel. It is based on the Hadoop ecosystem, using Apache Spark Streaming for streamed data and Apache Spark for batched data. Casandra is then used for query and storage. The data source consists of one minute summary records containing counts per dimension group. The dimension groups included time stamp, Device, APN, OS,FQDN and DNS transaction flag. The data was obtained through DNS traces done on the MNO. Anomaly free intervals are manually selected and rearranged. It is not clear from the article what the process for manually removing the anomalies is, this aligns with the approach taken in this study, where unsupervised learning approaches are used to remove anomalies. Synthetically generated anomalous data points, corresponding to the three predefined DNS anomalies were added. The three types of anomaly were observed in the MNO prior to model creation. These included short lived high intensity anomalies seen on the same devices/OS, low intensity anomalies lasting several days and short lived variable intensity anomalies for all devices on a specific APN. The "super learner" approach is stacking learning algorithm. A series of first level supervised learning algorithm (SVM with a linear kernel, decision trees, KNN, NN and naive Bayes) are used to produce the meta data input to the meta learners (linear regression, CART). The results proved marginally better than when compared to other ensemble techniques (Random Forest, Bagging Tree and AdaBoost Tree), and the first

level learners. This approach assumes that the type of anomalies to be identified are known up front. Only anomalies like the ones that are predefined will be detected by this approach.

Another study focusing on identifying anomalous DNS traffic is covered in [14]. Here the goal is to detect anomalies in popular applications such as YouTube and Facebook through identifying two manually identified anomalies in the DNS traffic profile. The study focuses on using decision trees as the main detection technique. These are compared to two statistical-based detection approaches namely Distribution-based Anomaly Detection (DAD) and Entropy-based Anomaly Detection, using an Exponentially Weighted Moving Average change detector (H-EMMA). Added to this five standard supervised ML-based approaches are considered for comparison purposes: Multi-Layer Perceptron Neural Networks, Naive Bayes, Locally-Weighted based Learning, Support Vector Machines, and Random Forest. The semi synthetic data used consisted of 10 minute level aggregations sourced from traces on DNS traffic. Six features are selected based on expert knowledge, including Device OS, Device Manufacturer, APN, FQDN of Remote Service, the Status of the DNS transaction and the time interval. The data points are then manually labeled and anomalous events are removed. The removal process is not perfect as the ground truth is not fully known. The data points are randomized, keeping the day type and time of day unaltered. Anomalies were manually generated and inserted into the data set making it possible to use supervised learning techniques. Once again, this approach also requires prior knowledge of the anomalies to be detected.

A study into mobile network subscriber activity analysis and subscriber anomaly detection is described in [15]. The aim of the study was to identify anomalies, the root cause of the anomaly as well as predicting anomalies. Anomalies were considered to be "abnormal or unusual behavior or activity patterns of user". The source data consisted of call data records (CDRs) for voice and SMS traffic which were stored in a Hadoop file store. The Voice and SMS activity in the CDRs were aggregated into one hour intervals and into geographical areas. The techniques used included K-means and Hierarchical Clustering for anomaly detection and Neural Networks for traffic predictions. Ground truth data was obtain to identify the cause of the anomalies. The approach managed to successfully identify anomalous subscriber activity including for example increased traffic due to a concert performed in one of the geographical areas. A concern with following this approach and using K-means as the anomaly detection algorithm is that as will be seen later, the data set we use consists of data with a much higher level of dimensionality (1000 compared to 4) with the many variables being highly correlated. Although this study took the time of day into account as one of the input parameters, it didn't take the sequential nature of time into account. This sequential nature of time was leveraged in [16]. Here an extended version of the incremental clustering algorithm, Growing Neural Gas (GNG), namely Merge Growing Neural Gas algorithm (MGNG) takes into consideration the history of input data. This is an incremental clustering/profiling model that incorporates time as a dimension. Anomalies are considered to be performance counters, performance counter combinations or sequences that are not normal. That is abnormal states as well as abnormal state changes such as going from one normal state to another normal state that does not normally follow the first state. Staying in one normal state for too long is also considered an anomaly. The data used in the study is hourly 3G Cell level performance stats consisting of 6 randomly selected Counters. 3 weeks of data for two cells (one control and one test). In the first two week's worth of data neither cell sees degraded service. This constitutes one week of training data and one week of anomaly free test data. The last week's data contains anomalies on only the test cell. The study successfully demonstrated the models ability to identify unusual cell state changes. Once

again [16] requires portions of the input data to be anomaly free for the approach to be successful.
The use a One-class SVM to detect malicious network attacks by identifying chances in IP traffic profiles of an ISP in Luxembourg is discussed in [17]. The data set was synthesized by combining the IP flow records generated by the network routers (Netflow records), which were considered to be anomaly free, with IP flow records from the Flame website. These records represented a series of attacks. An average accuracy of 92% was achieved while keeping the false positive rate below 2%. As with the majority of the studies we have reviewed up to now, this approach also required prior knowledge of what anomalies are as well as anomaly free training data.
A study [18] based on data from China Mobile Cellular Network, Baiyin, Gansu, aims at identifying traffic patterns that deviate from the average. 3 weeks of performance management data was collected from one Radio Network Controller (RNC). This covered 80 Base stations (340 Cells). 582 Performance Attributes were included (only 1, user plane traffic volumes were discussed, implying that anomaly detection is done per KPI individually and doesn't take any interaction between KPIs into account). Data is aggregated on 3 levels, spatially (RNC, BS and Cell), temporal (15min and hourly levels) and then lastly into temporal groups based on a user plane volume threshold. Anomalies are considered to be outliers after seasonal and secular trends have been removed. No ground truth information is take into account. The values are compared to the averages of the data on the same aggregation dimension. To reduce the amount of false alarms at low traffic periods [18] proposes that data be aggregated using a volume threshold. It continues on this path of reducing granularity by using K-means to cluster Base Stations based on similarity of KPIs compared temporally. The Silhouette coefficient is used to evaluate the quality of the clustering results. A concern about the aggregation approach used here, which requires a predefined minimum threshold of user plane data to be exceeded before alarms are triggered, is that some fault conditions could result in the cessation or retardation of user plane traffic. Faults of this nature would not be identified.

Chapter 2

Feature Engineering and Data Mining

The first part of this chapter describes the source and structure of the data used in this study as well as the cleaning of the data. The second half of the chapter goes on to describe various methods used to identify and extract the anomalies in the the data with the aim of creating an anomaly free data set.

2.1 Data Source

The section describes the source, format, flow, aggregation and storage of the data used in this study. Figure 2.1 gives an overview of what is covered. This investigation was required to understand the content of the data as well to as identify potential data corruption and data completeness issues which themselves can be presented as anomalies.
The source of the data are the GGSN/SGW/PGW nodes in the mobile network. Each node has been configured to measure and record the activities it is involved in. This data is saved locally to a file referred to as a "Bulkstats" file. Each node generates a new "Bulkstats" file every 15 minutes to capture the data associated with that 15 minute interval. Each file is divided up into groups of counters referred to as schema. There is one schema for each major function performed by each node. For example the "RADIUS" schema holds all the counters associated with the RADIUS function while the "Diameter" Schema holds counters associated with the Diameter function.
Each schema holds two types of data. The first are referred to as the "Statistics" and the second are referred to as the "Key Variables" [19]. The statistics can be one of three types. The first is of type "counter", which increments with each event counted, until the counter limit is reached and the counter rolls-over back to zero. An example of this would be the count of the number of bytes going through an interface. The second is of type "Gauge", which gives an absolute value at a point in time. An example of this is the number of data session supported at one point in time. The last type is "information" which is used to differentiate sets of statistics. An example of this would be an IP address. The second type of data held in a schema are the "key variables". These identify the dimensions that the "Statistics" are associated with and vary from schema to schema. Table 2.1 on page 15 lists the number of "statistics" and "key variables" associated with each schema.
Both the "Statistics" and "Key Variables" are stored in one of 4 data types. These are int32, which is a 32 bit integer, which rolls over at 4,294,967,295, Int64 which rolls over at 18,446,744,073,709,551,615, Float, which include decimal points and lastly String, which is used to represent characters.
An ETL (extract, transform, load) process is performed by a collector. It collects the

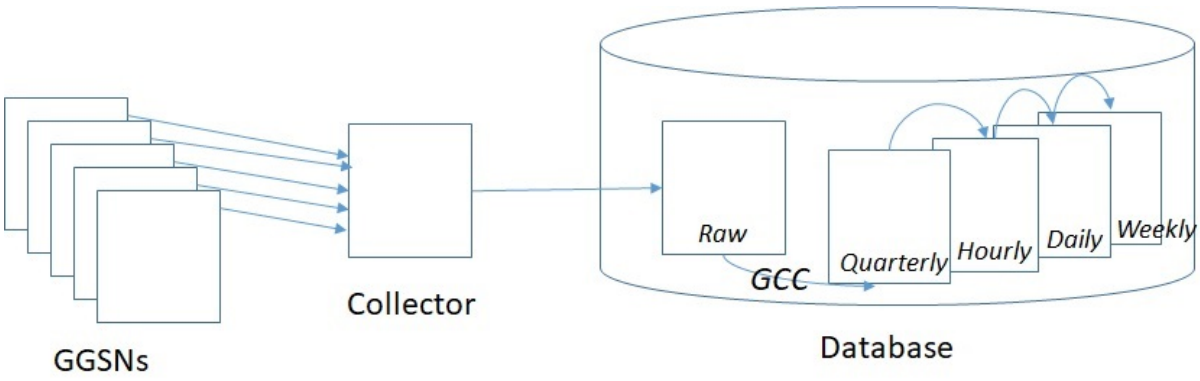


Figure 2.1: Flow of data from Generation through GCC and Aggregation to Storage.

"Bulkstats" files from the GGSN/SGW/PGW nodes and passes them on to the the database function. The Database function loads the data into "Raw" tables, joining all the schema from the individual nodes together. This results in there being a table or a set of tables per schema. Table 2.1 on page 15 lists the 26 schema.

As part of the loading process, the GGSN/SGW/PGW ID, the geographical region that the node is located in and the start time of the 15 min interval are added to each line of the table or tables. The data base function creates a "QTR" table associated with each "RAW" table by gauge counter correcting (GCC) the statistics of type "Counter". This process involves subtracting the counter value of the previous interval from the current interval to obtain the absolute value associated with that 15 minute period. The "statistics" of type "Gauge" are copied directly from the "Raw" tables to the "QTR" tables . The Database function is then responsible for aggregating data up in the time domain to hourly, daily, weekly, monthly and yearly tables. The database function is lastly responsible for deleting old data. The retention periods for each table type is given in table Table 2.2.

An investigation of the data flow reveals that there are several anomalies that arise in the data as a result of issues in processing of the data. These anomalies are described below, starting with the generation of the data on the GGSN/SGW/PGW nodes and ending with the aggregation of the data in the time domain. When generating the "statistics" of type "counter", it is expected that the counter value will roll over back to zero when it exceeds the maximum value that can be represented by the data types described above. Two scenarios have been identified where this is not the case. The first case presents itself when the process responsible for managing the counters restarts. When this happens all the counter increments for the associated 15 minute interval are reset. This results in counters being reset more frequently. This varies in frequency between the nodes. The nodes carrying more traffic have higher number of occurrences. The second case where the roll over occurs too frequently are in the cases where the data type used for the counter is not large enough to represent the increase in the statistic. Figure 2.2 on page 16 shows an example of the happening with counter G25M0C2. Sub-figure a shows the values for GGCT03 as a monotonically increasing function in the raw stats for all data points except for the first 3 which have experienced a roll over. Sub-figure b shows how the raw data is gauge counter corrected (GCC), effectively showing the counter values for GGCT03 (except for the first 3 data points). GGPS02 however carries more traffic, resulting in the counter rolling over in every 15 minute interval. Sub-figure b shows the corrupted for GGPS02 data after GCC.

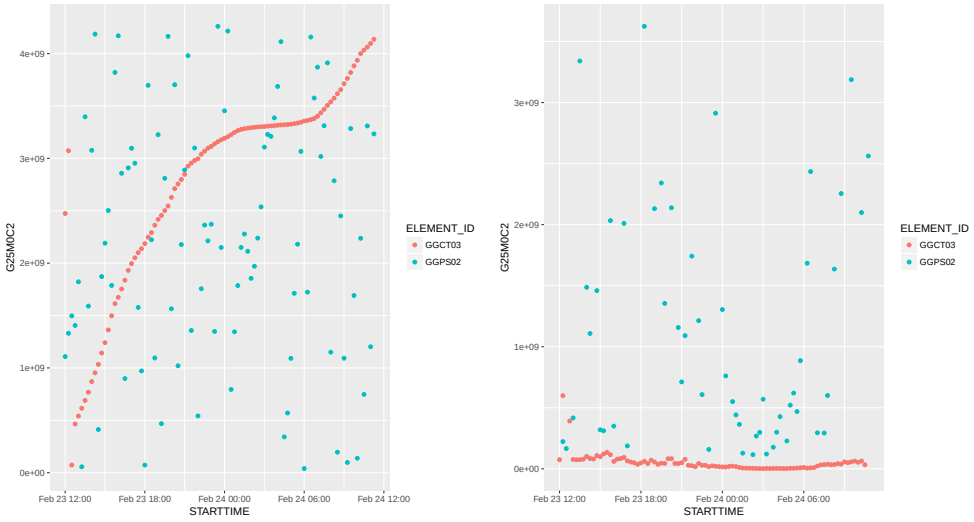
The next group of potential anomalies are introduced by the ETL process. Occurrences were observed where the collector function is interrupted. This results in missing data on

Table 2.1: Data “statistics” and “key variable” counts.

Schema	Tables	Sub Schema	Key Variables	Statistics
APN	4	4	3	309
APNQCI	1	1	2	16
CARD	3	3	1	136
DCCA	2	2	4	26
DCCASCH	1	3	2	123
DIAUSCH	1	1	7	44
DISCH	1	1	1	53
DPCA	3	4	4	41
ECS	12	12	0	1669
EGTPC	6	9	4	564
GTPC	5	5	6	276
GTPP	3	3	2	180
GTPU	2	2	4	91
IMSA	2	2	4	165
IPPOOL	1	1	8	13
LINK	1	5	2	180
P2PSCH	1	1	7	5
PDSN	15	22	7	2754
PES2A	2	25	5	488
PES2B	2	27	5	568
PES5S81	3	50	5	965
PGW	10	11	4	553
PORT	1	1	2	32
RADIUSGRP	1	1	7	78
RULEBASE	1	1	1	6
SGW	16	19	4	994

Table 2.2: Data retention periods.

Table type	Retention period
RAW	14 days
15 minutes	14 days
Hourly	90 days
Daily	400 days
Weekly	2 years
Monthly	5 years
Yearly	10 years



(a) Raw data. (b) Gauge counter corrected data.

Figure 2.2: Raw data showing roll overs on counter G25M0C2.

all or some of the nodes. A miss-alignment in the timing between the collector and the database results in examples where the statistics are double counted or missing for a subset of the nodes. Incorrect "statistics" definitions on the database function results in data being corrupted during the creation of the QTR table. Instances were found where some statistics of type "Gauge" were treated as type "Counter" and some of type "Counter" were treated as type "Gauge". Lastly some counters had the incorrect aggregation rule applied to them for the time domain aggregations. For example a counter measuring a percentage should be aggregated using a maximum or average function and not a sum function, and likewise a counter measuring the count of packets, should be aggregated using sum and not average or maximum.
With the goal of reducing the impact of data corruption on the subsequent anomaly detection methods the following steps were taken. The counter type definitions were audited and corrected. The timers controlling the database loading from the collectors were adjusted to ensure no missing or duplicated stats. The use of the incorrect data types resulting in roll overs were logged with the equipment vendor (the fixes for these would only be made available in the following software release). The aim was to address the rest of the anomalies introduced in the data flow through filtering them out. This meant that the QTR tables would have to form the basis of the study as the tables with the higher time domain aggregation had already aggregated the missing and corrupted data in with clean data.
The study focuses on the data contained in the SGW Schema.

2.2 SGW Data Description

The SGW schema contains 994 "statistics", four Key Variables and the added start time and SGW element dimensions. The data sample extracted was for a 14 day interval from 2018-02-24 09h00 to 2018-03-10 08h15 and consisted of 28161 observations. The key variables include The VPN_id, VPN_NAME, SERV_ID and SERV_NAME. Table 2.3 on page 17 shows the break down of these per GGSN/PGW/SGW node. Each node only has one VPN_ID and one SERV_ID and which map to the VPN_NAME of "SGW" and SERV_NAME of "SGW_SVC" respectively. The key variables therefore do not indicate any sub dimension and can be ignored.

Table 2.3: SGW Schema Key variables.

SGW_ID	VPN_ID	VPN_NAME	SERV_ID	SERV_NAME
CSGNMT01	6	SGW	8	SGW_SVC
GGCF01	4	SGW	9	SGW_SVC
GGCF02	5	SGW	9	SGW_SVC
GGCT01	14	SGW	9	SGW_SVC
GGCT03	7	SGW	9	SGW_SVC
GGCT04	7	SGW	9	SGW_SVC
GGDM01	4	SGW	9	SGW_SVC
GGDM02	3	SGW	3	SGW_SVC
GGDN01	9	SGW	9	SGW_SVC
GGDN02	9	SGW	9	SGW_SVC
GGDN03	3	SGW	3	SGW_SVC
GGJF01	4	SGW	9	SGW_SVC
GGMT01	20	SGW	11	SGW_SVC
GGMT03	7	SGW	9	SGW_SVC
GGMT04	7	SGW	9	SGW_SVC
GGPR01	4	SGW	9	SGW_SVC
GGPS02	18	SGW	9	SGW_SVC
GGPS03	7	SGW	9	SGW_SVC
GGPS04	5	SGW	3	SGW_SVC
NFV1-GGMD01	5	SGW	9	SGW_SVC
NFV1-GGPR02	3	SGW	9	SGW_SVC

The SGW nodes are deployed across 4 regions as shown in Table 2.4

Table 2.4: Regional location of GGSN/PGW/SGW nodes.

CTN	DBN	JHB	PTA
GGCF01	GGDM01	GGJF01	GGPR01
GGCF02	GGDM02	GGMT01	GGPS02
GGCT01	GGDN01	GGMT03	GGPS03
GGCT03	GGDN02	GGMT04	GGPS04
GGCT04	GGDN03		NFV1-GGPR02
	NFV1-GGMD01		

The software program R [20] was used for all subsequent data analysis and model building. The following processes was used to clean the SGW data. The variables consisting on only zero values were removed first. The variables with a high percentage of N/A entries were then removed. Next, the observations with an N/A entry were removed. The threshold used for the percentage of N/As per variable, affected how many observations were removed in the final step. This threshold was selected to minimize the amount of data removed from the original data set. Lastly it was observed that some of the nodes did not support the SGW function. The observations for these nodes only contained zeros and were removed.

Out of the original 994 variables, 636 contained only zeros, leaving 358. The remaining variables consisted of only positive integers and N/A. Table Table 2.5 on page 18 shows the number of variables with the same number of N/A present. For example there were 670

Table 2.5: Variables count per N/A number.

Variable Count	N/A Count
670	0
6	1
17	2
28	3
11	4
28	5
2	6
5	7
24	8
93	9
104	10
3	11
1	81
1	137
1	139
1	584
1	8075
1	8997

variables with 0 N/A present and there was 1 variable with 8997 observations containing N/A. All variables containing more than 11 N/A were removed from the data set. This left 349 of the original 994 variables. After removing observations containing N/A 28148 of the original 28161 observations remained. 9 of the nodes didn't support the SGW function were removed, leaving 12 nodes. The observations remaining after the removal of these nodes was 16070. Table 2.6 on page 18 shows that in the final SGW data set, the number of observations per nodes is balanced.

Table 2.6: Observations per GGSN/SGW/PGW in the final SGW Data.

SGW_ID	Observations
GGCF01	1342
GGCT03	1342
GGCT04	1342
GGDM01	1337
GGDN01	1342
GGDN02	1337
GGJF01	1341
GGMT03	1342
GGMT04	1332
GGPR01	1336

The activity supported by the SGW function is expected to be periodic, following the activity of the mobile subscribers. Variable G19M4C49 represents the down link bytes on the S1U interface of the SGW. Figure 2.3, Figure 2.4 and Figure 2.5 show the following characteristics of the data volumes supported by the SGWs.

- The traffic volumes follows a daily cycle, with the lowest point being in the early

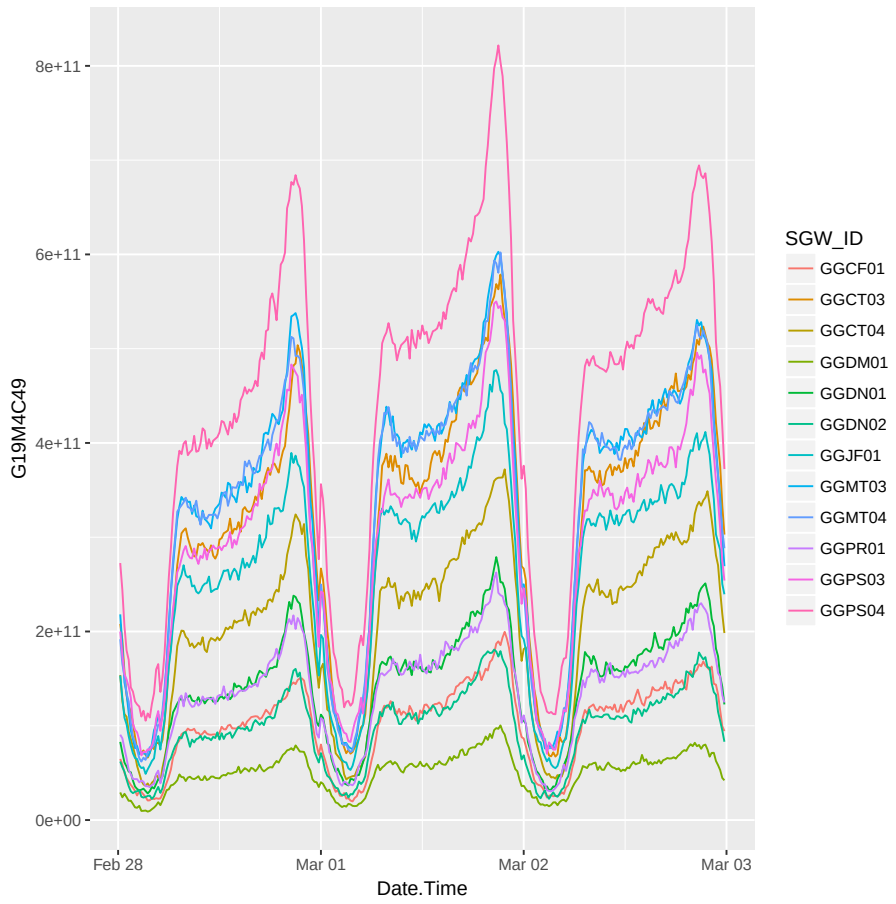


Figure 2.3: S1U Down link Bytes per SGW.

hours of the morning and the highest being at 21h00

- The traffic volumes supported by the different SGW nodes varies with GGPS04 carrying the most traffic and GGDM01 the least
- The traffic volumes on the first of the month are higher than the traffic on the last day of the month
- The traffic volumes on a Saturday and a Sunday stays higher into the early hours of the morning compared to a weekday
- The traffic volumes on a Sunday stays lower for longer in the morning than the other 6 days of the week
- The difference in traffic volumes between the mid-morning and late evening in lowest on a Saturday.

There are two dips seen on the day 7 and day 4 of Figure 2.3. These caused by there being a missing observation for GGMT04.
As the variables are measurements of the various activities supported by the node, it is expected that there will be high levels of correlation between the variables. For example, in the early hours of the morning, when the majority of subscribers are sleeping, very few subscribers will be sending or receiving data, this means that the number of packets sent, bytes send, packets received, bytes received, will be lower for all 9 Quality of Service Class Indicator groups. This means that all 36 counters measuring these will have low values. As people start waking up and using data all of these counters will start increasing in

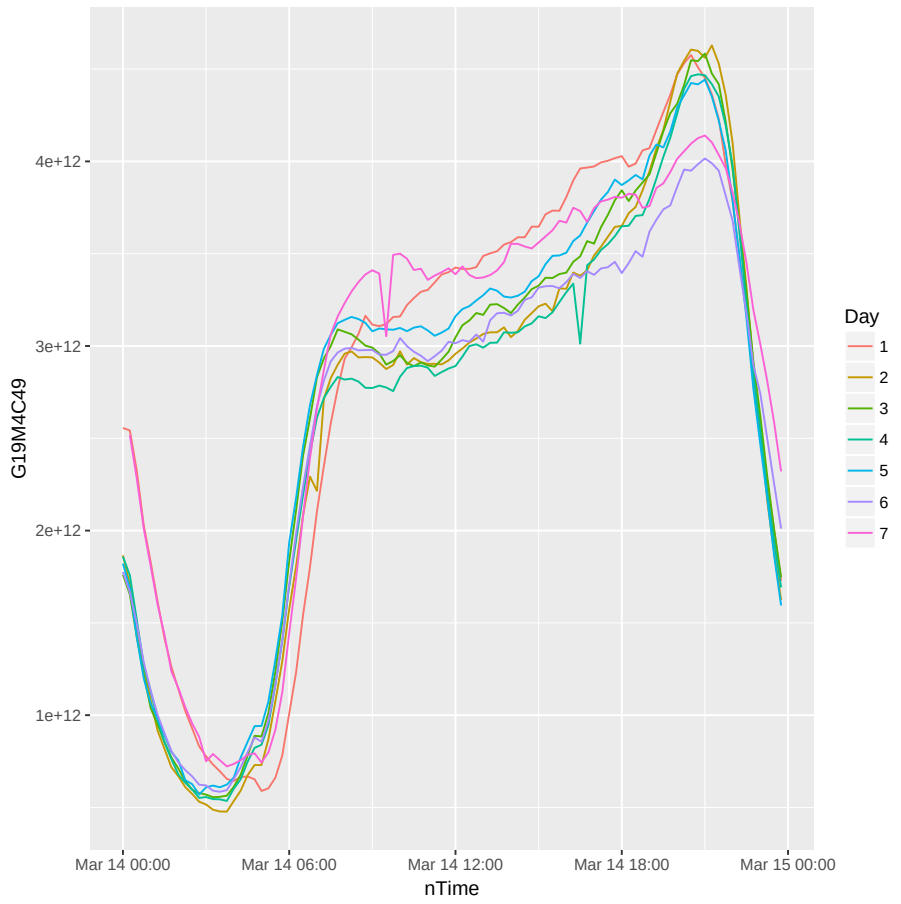


Figure 2.4: Day of week comparison.

sync. As there are too many variables to show in a correlation matrix, the correlation matrix is represented in a bitmap format where correlations of 1 are shown as black and -1 as white. Everything in between is shown as in varying shades of gray. Figure 2.6 shows the high levels of correlation.
Figure 2.7 and Figure 2.8 show a summary of the SGW variables. There is not an even distribution of the values, with one grouping of low values dominating the results. A log scale on the x axis is used to show the distribution more clearly.

2.3 Methods

The approach taken is to prepare the data set so that it can ultimately be used in a semi-supervised learning approach. This required the identification and removal of outliers from the original raw data. To achieve this, Principal Component Analysis (PCA) is first used to reduce the number of variables. This data is then presented to 4 anomaly detection techniques, namely K-nearest neighbours, One Class Support Vector Machines, Density-Based Local Outlier Factor and Multivariate Gaussian Distributions. The results are used to identify and remove the observations most likely to be anomalies within the raw data set. The resulting data set forms the input to a semi-supervised learning approach.

2.3.1 Dimensionality Reduction

As shown in Figure 2.6, the SGW data is highly correlated and will therefore benefit from dimensionality reduction through PCA. PCA was first introduced in [21]. The approach

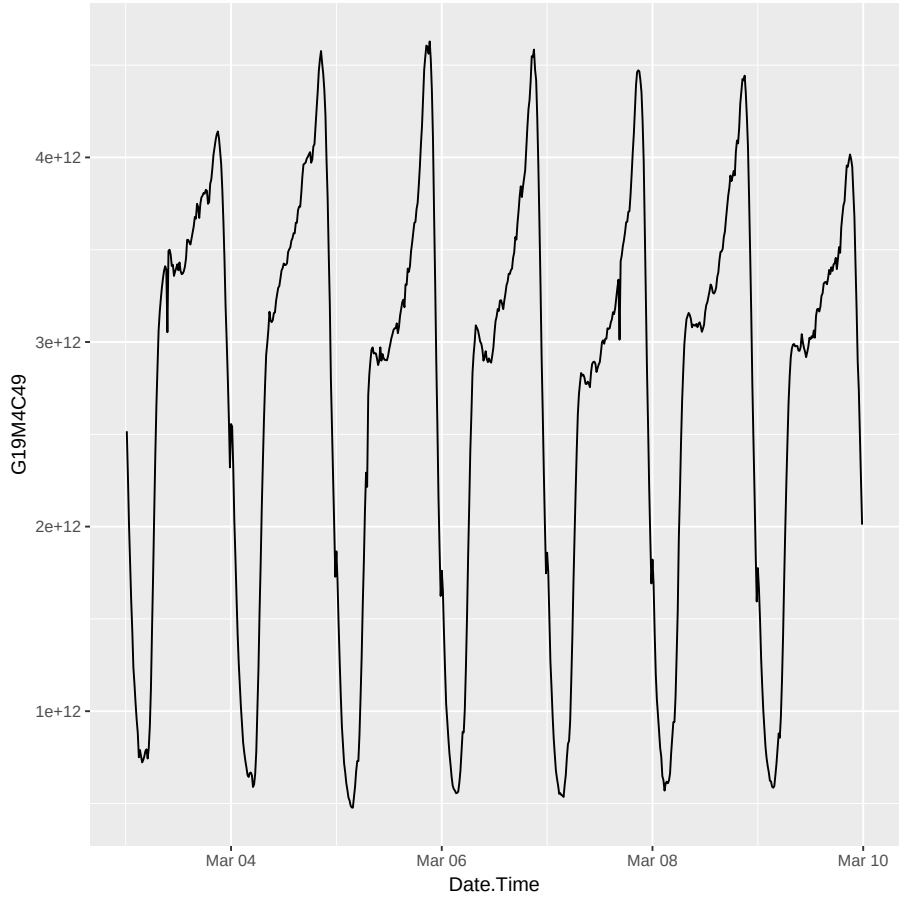


Figure 2.5: Total network S1U bytes down over 7 days.

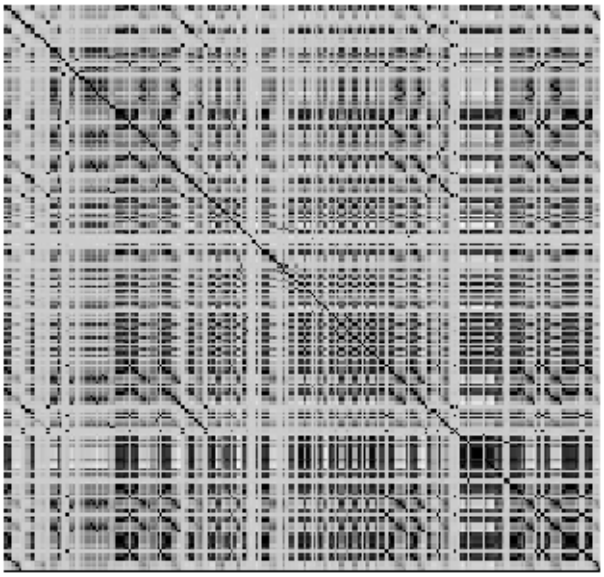
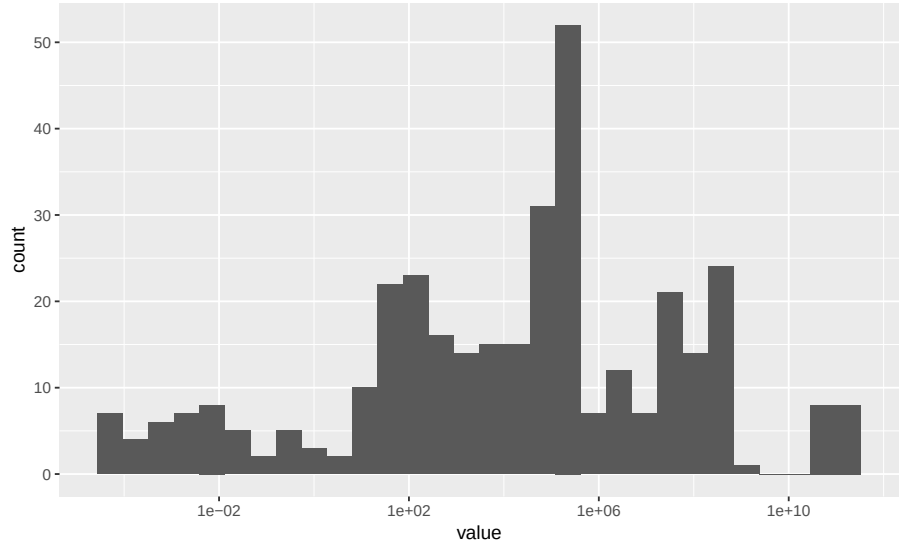
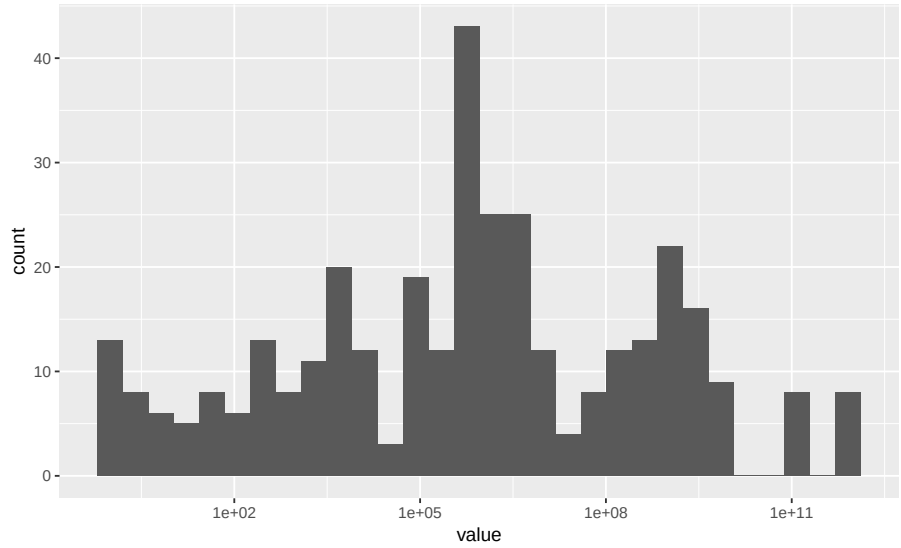


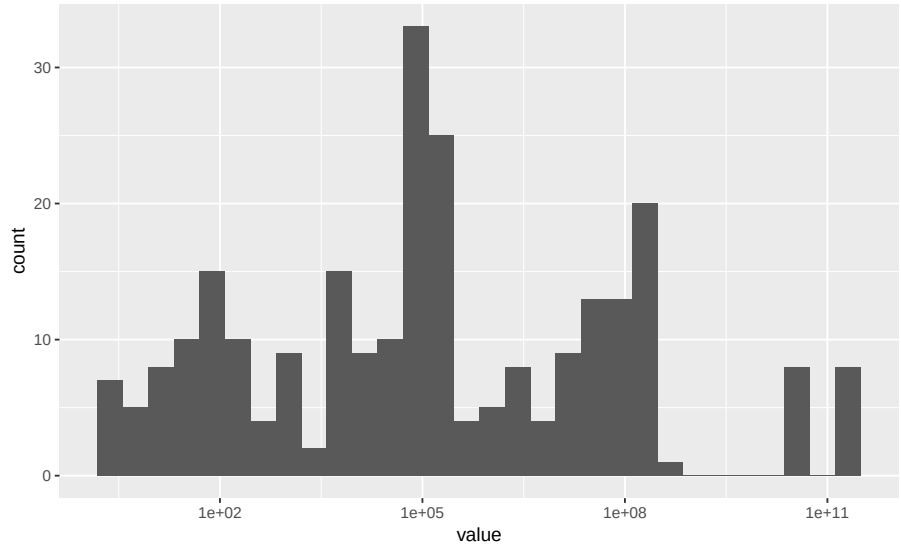
Figure 2.6: Flow of data from Generation through GCC and Aggregation to Storage.



(a) Histogram of the means of the SGW variables.

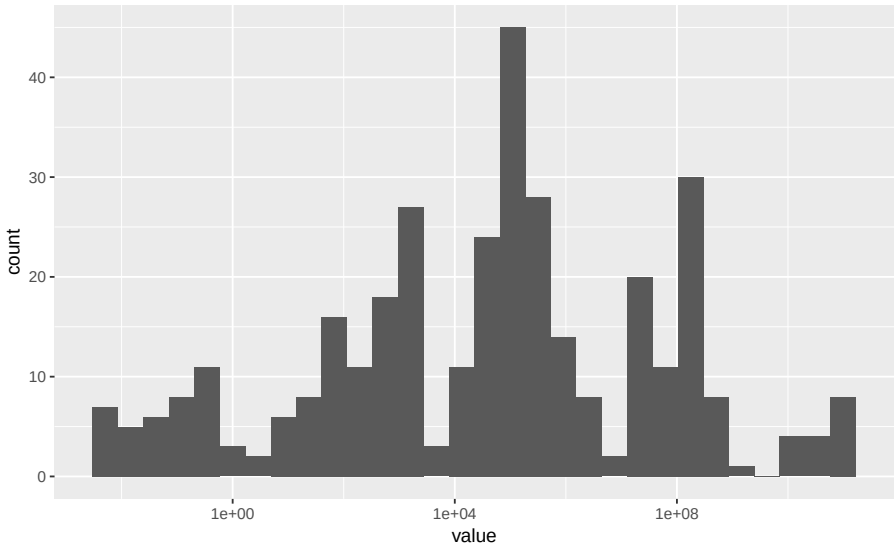


(b) Histogram of the maxes of the SGW variables.

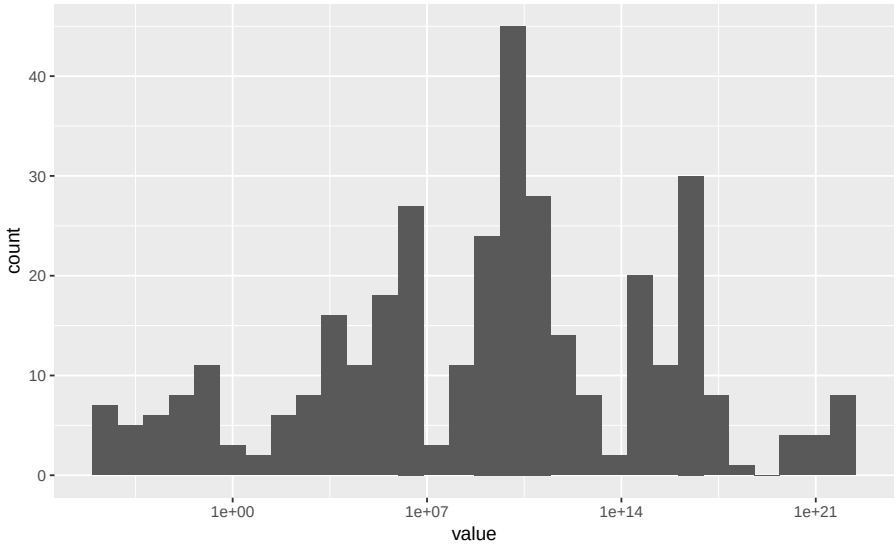


(c) Histogram of the medians of the SGW variables.

Figure 2.7: SGW data summary.



(a) Histogram of the standard deviation of the SGW variables.



(b) Histogram of the variances of the SGW variables.

Figure 2.8: SGW data summary.

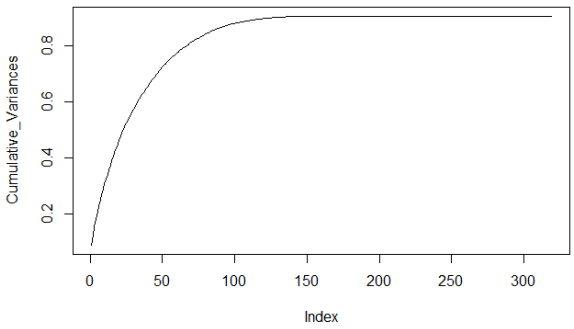


Figure 2.9: Cumulative Variation explained by all 349 principal components.

removes redundancy in the data by creating new independent variables (principal components) that are linear combinations of the original variables. The first principal component explains the most variation in the data, with each subsequent principal component explaining less and less. The goal is to reduce the number of dimensions/ variables used in the subsequent anomaly detection techniques employed to remove anomalies from the data as preparation for the semi-supervised learning approach. Added to this the impact of the "curse of dimensionality" will be reduced in the subsequent analysis.

The variation explained by the first 3 principal components of the SGW data is 8.55%, 5.52% and 3.12% respectively, making up 17.2% together. The first 60 principal components explain 81.67% of the variation and the first 100 are needed to explain 95.23%. Figure 2.9 on page 24 shows the cumulative variation explained by all 349 principal components.

The plot of the first two principal components in Figure 2.10 shows potential clusters based on the time of day with observations occurring at the same time of day being grouped together. Figure 2.11 shows that the observations generated by each node also form clusters.

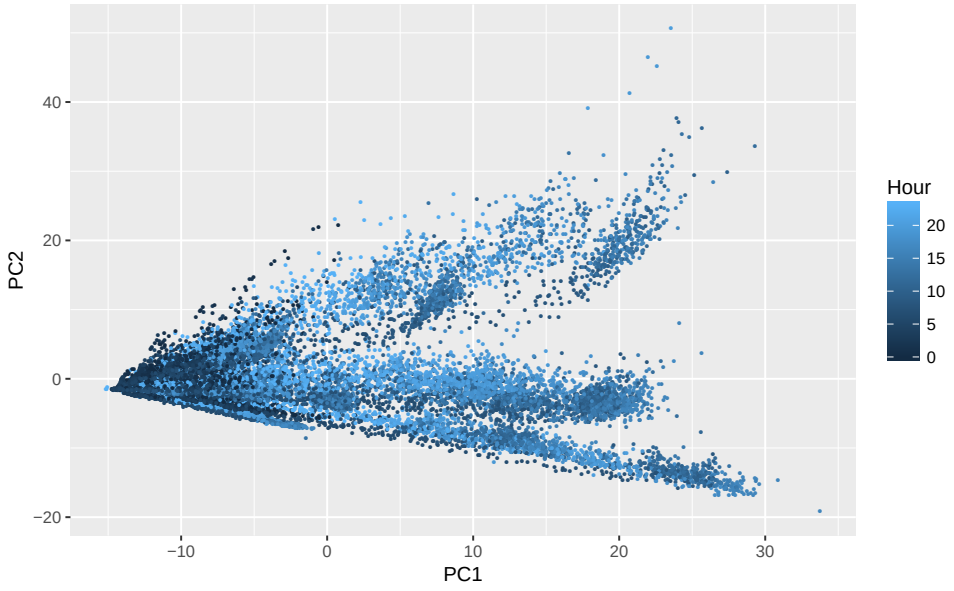


Figure 2.10: First and Second principal components colored per hour.

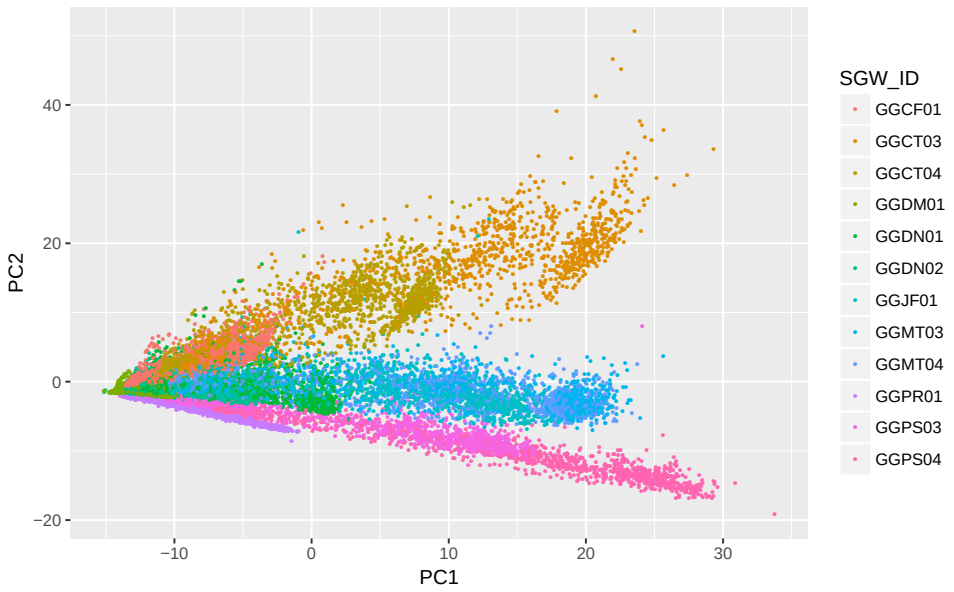


Figure 2.11: First and Second principal components per node.

When faceting the data further per node and day of the week, as shown below, outliers based on the first two principal components start to become apparent. Examples are seen on day one for GGCT04 and day 3 for GGDN01.

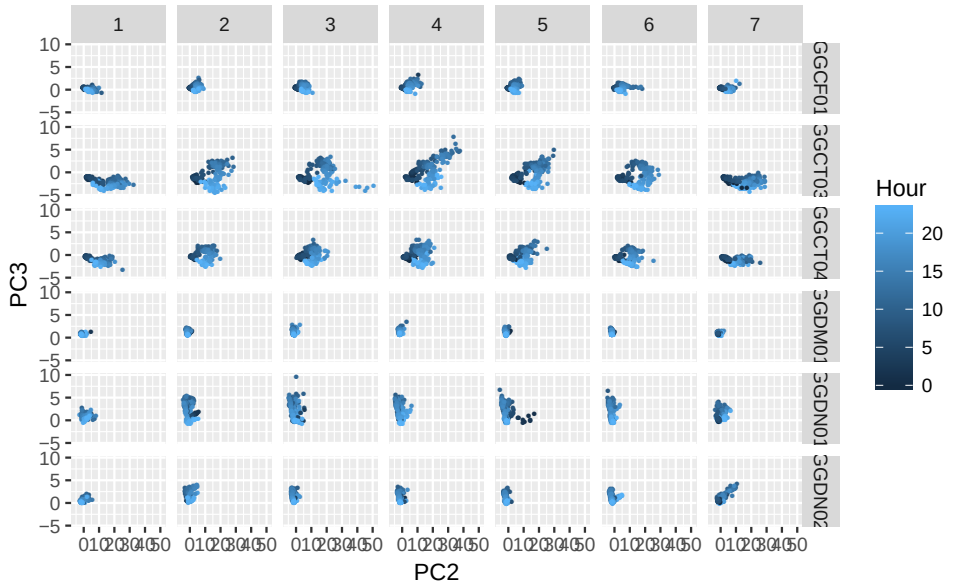


Figure 2.12: First and Second principal components per node faceted per day.

Focusing in on one of the nodes, GGJF01 and faceting on day of week and hour of the day, it is possible to see more clusters based on these dimensions.

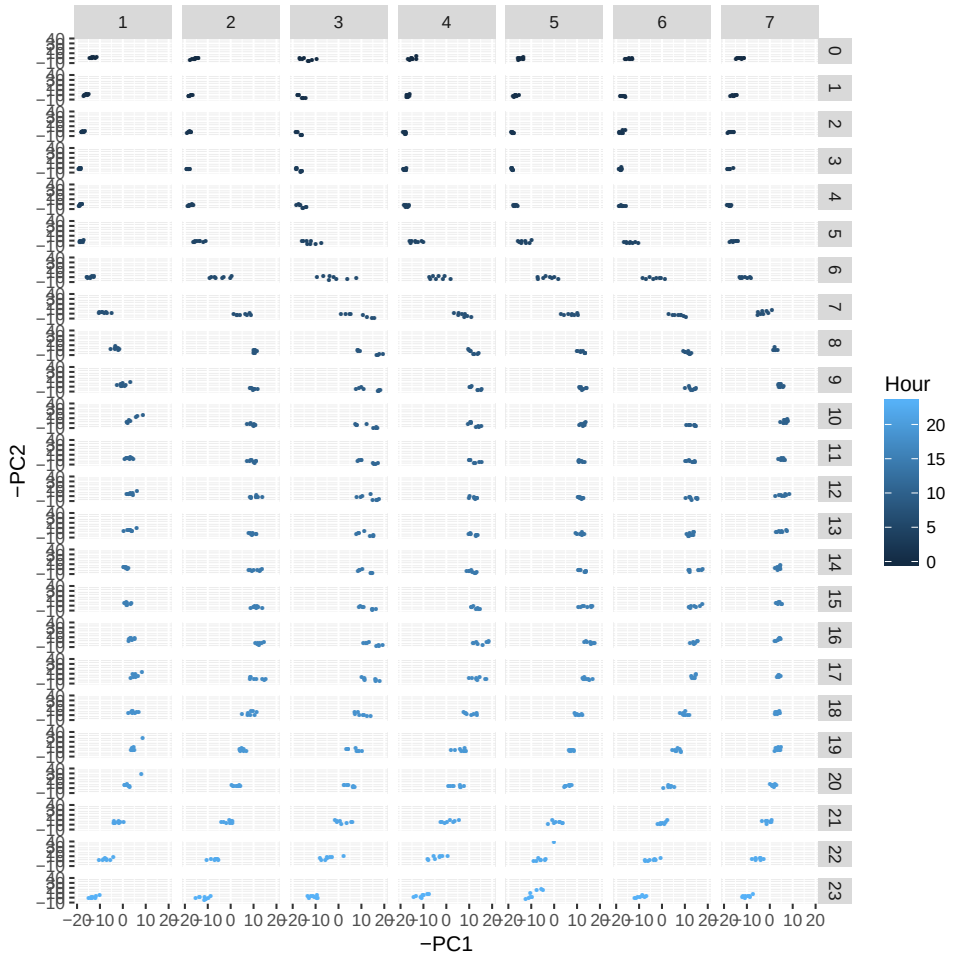


Figure 2.13: First and Second principal components for GGJF01 node faceted per day.

Figure 2.14 once again shows the same dimensions for GGJF01, but instead of being based on the original PCA data which was done on the entire data set consisting of all the nodes, this shows the result of a separate PCA done on a subset of this data, consisting of only the GGJF01 observations. It is apparent that the results are similar with similar cluster and outliers, but not identical. For example both show the outliers on day one for hours 19 and 20, but it is less obvious in the original PCA data. The original PCA data doesn't show the anomaly evident on day 4, hour 2.

2.3.2 K-Nearest Neighbours

Prior to [6], Knorr and Ng's paper titled "Algorithms for Mining Distance-Based Outliers in Large Datasets", outlier detection techniques were limited to statistically based approaches. An alternative distance based approach is proposed in [6]. The aim was to overcome the limitations of the statistical based approaches which included their inability to support high dimensional data sets and the need to have an understanding of the nature of the distribution of the data set. Knorr's definition of an outlier is "An object O in a dataset T is a $DB(p, D)$ (distance based) outlier if at least fraction p of the objects in T lies greater than distance D from O.". The algorithm for identifying outliers takes each observation and counts the number of neighbours within the distance D, if there are less than $T - pT$ it is classified as an outlier. This resulted in an algorithm complexity of order $O(kN^2)$. Two shortcomings of this approach include that no ranking of the outliers is provided as well as the need for the user to decide on the distance D through trial and error. These shortcomings were addressed in [22]. This proposed calculating the distance

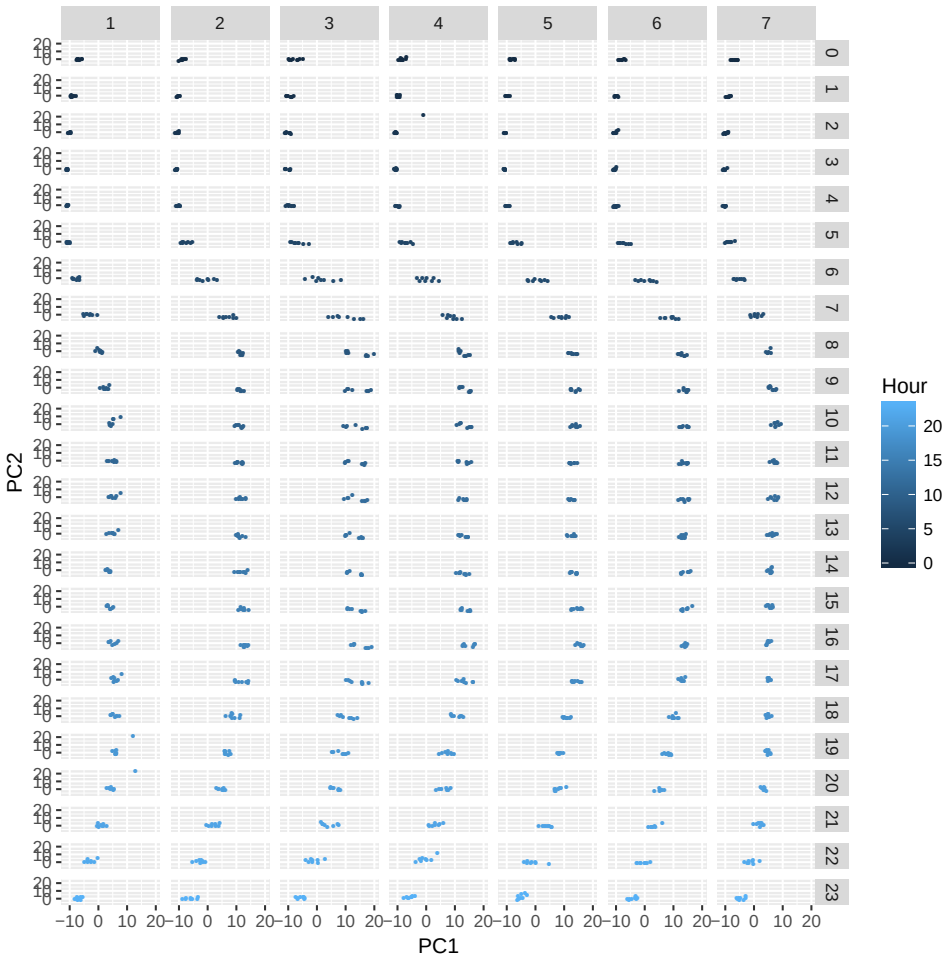


Figure 2.14: First and Second principal components based only on GGJF01 data faceted per day.

to the k^{th} furthest neighbour and then declaring the x points with the highest distances as the outliers. The outliers' distances to their k^{th} neighbour was used for ranking the level of each outlier. An alternative to using the distance to the k^{th} nearest neighbour is proposed in [5], where the outlier score, referred to as the "weight" is calculated as the average distance to the k nearest neighbours.
These KNN approaches identify global anomalies but not local anomalies. The choice of k will impact the results, with low values resulting in the density estimation being potentially unreliable and high values can potentially result in density estimations being too coarse [4]. According to [4] no formalized approach for defining the best value k exists upfront, that a range of values between 10 and 50 are investigated and compared.
"The strength of those algorithms stem from the fact that they are inherently unsupervised and have an intuitive criteria for detecting outliers. Their limitations include the quadratic computational complexity and a possible incorrectness when handling high dimensional data." [23]
The first KNN analysis done on the SGW data set was done on the first 100 principal components (explaining 95% of the variation in the data). Both the k^{th} nearest neighbour and the weight based approaches were used. Five different values for k, being 10, 20, 30, 40 and 50 were used. The top 100 anomalies identified by each of the ten techniques are compared in Table 2.7. The anomalies identified by the 10 different approaches are similar with the average similarity being 92.5%. The biggest difference between the anomalies identified was between the approach using the weight of the 10 nearest neighbours and the approach using the 50^{th} nearest neighbour. Here 82 of the top 100 anomalies were common to both approaches. The number of unique anomalies identified by all ten approaches was 124.

Table 2.7: Comparison of KNN variations, showing matching outlier count per pair.

	50th NN	40th NN	30th NN	20th NN	10th NN	Weight of 50	Weight of 40	Weight of 30	Weight of 20	Weight of 10
50th NN	100	99	98	95	84	95	94	92	87	82
40th NN	99	100	99	96	85	96	95	93	88	83
30th NN	98	99	100	97	86	97	96	94	89	84
20th NN	95	96	97	100	89	98	97	97	92	86
10th NN	84	85	86	89	100	89	90	92	94	90
Weight of 50	95	96	97	98	89	100	99	97	92	87
Weight of40	94	95	96	97	90	99	100	98	93	88
Weight of 30	92	93	94	97	92	97	98	100	95	89
Weight of 20	87	88	89	92	94	92	93	95	100	94
Weight of 10	82	83	84	86	90	87	88	89	94	100

As discovered in the previous section covering dimensionality reduction, the anomalies need to be considered in context. To move from contextual anomaly detection to point anomaly detection, the data set was broken up into subsets. Various combinations of subsets were investigated. The first subsets were based on node. Here the KNN values were calculated for each node separately. Each of the ten KNN approaches were used and the top 100 of each of these approaches resulted in 112 unique anomalies. The matrix showing the number of common anomalies identified per KNN approach pair is shown in Table 2.8. The anomalies identified by the 10 different approaches are more similar than the first approach with the average similarity being 95.6%. The biggest difference between the anomalies identified was between the approach using the weight of the 10 nearest neighbours and the approach using the 50^{th} nearest neighbour. Here 89 of the top 100 anomalies were common to both approaches.
The approach using the node based grouping yielded 4 anomalies not identified in the first approach that considered all data points together. Table 2.9 shows these 4 new observations as well as the original and new KNN average ranking.
Table 2.10 shows the average rankings using both the node based approach and the original

Table 2.8: Comparison of KNN variations, showing matching outlier count per pair. Based on node groupings.

	50th NN	40th NN	30th NN	20th NN	10th NN	Weight of 50	Weight of 40	Weight of 30	Weight of 20	Weight of 10
50th NN	100	97	97	95	95	95	95	95	94	89
40th NN	97	100	99	97	97	96	96	96	95	90
30th NN	97	99	100	98	98	97	97	97	96	91
20th NN	95	97	98	100	98	97	97	97	96	91
10th NN	95	97	98	98	100	98	98	99	98	93
Weight of 50	95	96	97	97	98	100	100	99	99	94
Weight of 40	95	96	97	97	98	100	100	99	99	94
Weight of 30	95	96	97	97	99	99	99	100	99	94
Weight of 20	94	95	96	96	98	99	99	99	100	95
Weight of 10	89	90	91	91	93	94	94	94	95	100

Table 2.9: New observations ranked in the top 100 using the node based grouping.

Observation	Original KNN Ranking	Node based KNN Ranking
641	155,5	121,6
5007	136,5	111,6
5261	120,6	97,5
5462	134,3	119,2

non grouped approach for all observations that the original approach had top 100 ranks for but the node based approach didn't.

Table 2.10: Observations not ranked in the top 100 using the node based grouping.

Observation	Original KNN Ranking	Node based KNN Ranking
1431	116,6	111,7
4238	150,3	170,0
4262	141,9	162,2
5272	94,5	112,3
5891	99,1	113,0
5952	108,6	131,3
6203	123,3	141,6
6371	114,9	125,9
8497	130,1	141,5
9666	118,0	122,7
11471	115,7	135,9
12723	116,4	135,6
12783	101,9	120,5
15242	113,8	118,7

The second subsets analyzed were based on node and time of day. The time of day dimension represented the 15 minute interval associated with the observation. Here the KNN values were calculated for each node and time of day separately. The number of observations per subset only numbered 14. For this reason the number of nearest neighbours used was changed from 10, 20, 30, 40 and 50 to 2, 4, 6, 8 and 10. Each of the ten KNN approaches were used and the top 100 of each of these approaches resulted in 104 unique anomalies. The matrix showing the number of common anomalies identified per KNN approach pair is shown in Table 2.11. The anomalies identified by the 10 different approaches are even more similar than the the previous two approaches with the average similarity being 98.3%. The biggest difference between the anomalies identified

was between the approach using the weight of the 2 nearest neighbours and the approach using the 10^{th} nearest neighbour. Here 96 of the top 100 anomalies were common to both approaches.

Table 2.11: A Comparison of KNN variations, showing matching outlier count per pair. Based on node/time groupings.

	10th NN	8th NN	6th NN	4th NN	2nd NN	Weight of 10	Weight of 8	Weight of 6	Weight of 4	Weight of 2
10th NN	100	99	99	98	98	99	98	98	97	96
8th NN	99	100	100	98	98	99	98	98	97	96
6th NN	99	100	100	98	98	99	98	98	97	96
4th NN	98	98	98	100	99	99	100	100	99	98
2nd NN0	98	98	98	99	100	99	99	99	99	98
Weight of 10	99	99	99	99	99	100	99	99	98	97
Weight of 8	98	98	98	100	99	99	100	100	99	98
Weight of 6	98	98	98	100	99	99	100	100	99	98
Weight of 4	97	97	97	99	99	98	99	99	100	99
Weight of 2	96	96	96	98	98	97	98	98	99	100

The approach using the node/time based grouping yielded 5 anomalies not identified in the first two approach that considered all data points together and the grouping per node. Table 2.12 shows the three new observations not observed in the original analysis and Table 2.13 shows the four observation not observed in the analysis of the node based approach.

Table 2.12: New observations ranked in the top 100 using the node/time based grouping not seen in the original grouping.

Observation	Original KNN Ranking	Node based KNN Ranking	Node/time base KNN ranking
641	155,5	121,6	90
3938	167,4	188,7	101
4729	127,9	147,7	112,2

Table 2.13: New observations ranked in the top 100 using the node/time based grouping not seen in the node based grouping.

Observation	Original KNN Ranking	Node based KNN Ranking	Node/time base KNN ranking
3938	167,4	188,7	101
4238	150,3	170	90
4262	141,9	162,2	86,5
4729,0	127,9	147,7	112,2

Table 2.14 shows the average rankings using the node/time based, the node based and the original non grouped approach for all observations that the original approach and node based approaches had top 100 ranks for but the node/time based approach didn't.

2.3.3 One-Class Support Vector Machine (OCSVM)

Support vector machines (SVMs) are designed as binary classifiers, used to classify data into one to two possible classes. At the heart of the SVM classifier is a separating hyperplane. The distance that an observation is from the hyperplane gives insight into the confidence of the resulting classification. The further the observation is from the hyperplane the more likely the classification given is correct.

Table 2.14: Observations not ranked in the top 100 using the node/time based grouping.

Observation	Original KNN Ranking	Node based KNN Ranking	Node/time base KNN ranking
296	102,3	107,2	118,8
1431	116,6	111,7	126,7
2556	110	121,6	151,1
3995	91,9	100,2	119,2
4681	83,8	102,3	107,4
5007	136,5	111,6	112,3
5261	120,6	97,5	116,9
5272	94,5	112,3	132,1
5462	134,3	119,2	114,3
5743	110,4	116	146,4
5891	99,1	113	118,7
5952	108,6	131,3	137,6
6203	123,3	141,6	143,3
6336	82,1	100	108,6
6371	114,9	125,9	141,9
8057	103	101,1	108
8497	130,1	141,5	161,7
9666	118	122,7	149
10730	106,9	111,7	134,7
11471	115,7	135,9	144,2
12128	109,9	113,9	145,1
12723	116,4	135,6	159,8
12783	101,9	120,5	137,8
13338	113,6	102,6	107,6
15242	113,8	118,7	142,5
15348	90,3	91,9	104,9

The initial work leading up to support vector machines was done by Vapnik and Lerner in [24] with the concept of a maximal margin classifier. Here a hyperplane with the largest possible margin separating two linearly separable classes is identified. Single observations impact the position of the separating hyperplane indicating that overfitting is possible. In [25] the support vector network is introduced to overcome the requirement that the two data classes be perfectly separable. It introduces the concept of the soft margin. The soft margin tolerates training observations to be on the incorrect side of the margin and on the incorrect side of the separating hyperplane. Slack variables are given to these observations and a tuning parameter is used to control the amount and degree of observations that violate the margin. The higher the value to the tuning parameter the more tolerant the model is to observations violating the margin. The support vector machine introduced in [26] takes it beyond the support of linear boundaries to non-linear boundaries. It does this by enlarging the feature space, introducing non-linear components, in a specific, computational efficient way using kernels.

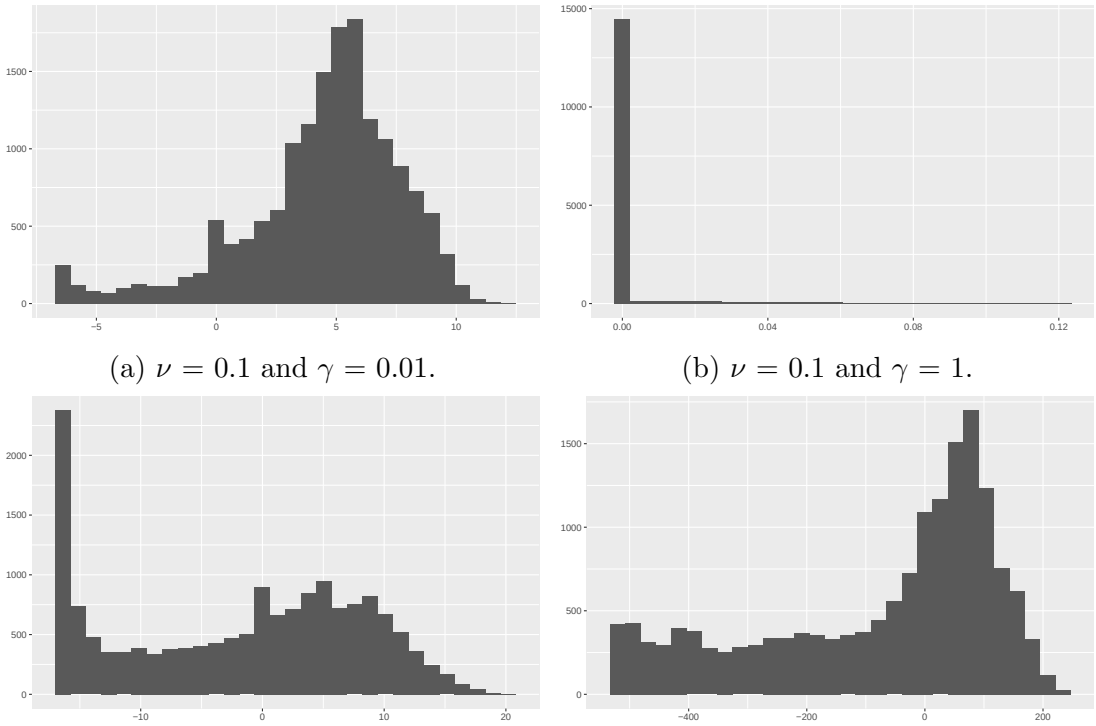
These 3 classifiers are supervised learning approaches. A one-class support vector machine was introduced in [27] with the aim of transferring this approach to the unsupervised domain. Here the data points are separated from the origin in the projected features space by a hyperplane. A tuning parameter ν is set for the soft margin to handle any anomalies correctly. With it setting the upper bound of the fraction of outliers and the lower bound on the percentage of support vectors used. Each point is assigned a decision value, with negative values being assigned to observations on the origin side of the hyperplane and positive values to the others. The magnitude of the assigned value is proportional to the distance from the hyperplane. Outliers will therefore be assigned high negative values.

A second one-class support vector machine approach was introduced in [28] where instead of using a hyperplane, the use of a hypersphere is used. According to [23] the one-class support vector machine is sensitive to outliers, with the outliers shifting the decision boundary. Two methods to reduce the impact of outliers on the decision boundary are introduced in [23]. The first is the robust one-class support vector machine, which allows observations far from the data set centroid to have large slack values. The result of this is the shifting of the decision boundary towards the normal observations. The second is the η one-class support vector machine. A variable η is added, indicating the estimation of a points normality. η values of zero indicate an outlying observation. Observations with values of zero are not included in the process of learning the decision boundary. The resulting decision boundary is only influenced by normal points.

In this study the one-class support vector machine introduced in [27] is used. The associated hyper parameters are ν, as described above, and the choice of Kernel, with its own associated hyper parameters. This was implemented using the interface to libsvm using the r package e1071. As the data set consists of unlabeled data, the hyper parameter search could not be done by checking the accuracy of classification. Instead the distributions of the of the resultant decision values where compared. Various hyper parameters were checked, and those that produced decision values with distributions grouping most observations above zero with a large separation to a few observations with large negative numbers were investigated further. Three such iterations were completed.

The first set of hyper parameters investigated included the radial basis function as the kernel with $\gamma \in \{0.01, 0.05, 0.1, 0.5, 1\}$ against all combinations of $\nu \in \{0.01, 0.05, 0.1, 0.5, 1\}$. A selection of the resultant histograms are shown in Figure 2.15. The results obtained with $\nu = 0.1$ and $\gamma = 0.01$ shown in sub-figure *a* met the selection criterion the best.

The second set of hyper parameters investigated continued with the kernel set to the



(a) $\nu = 0.1$ and $\gamma = 0.01$.

(b) $\nu = 0.1$ and $\gamma = 1$.

(c) $\nu = 0.5$ and $\gamma = 0.05$.

(d) $\nu = 0.5$ and $\gamma = 0.01$.

Figure 2.15: Resultant distributions in first hyper parameter search.

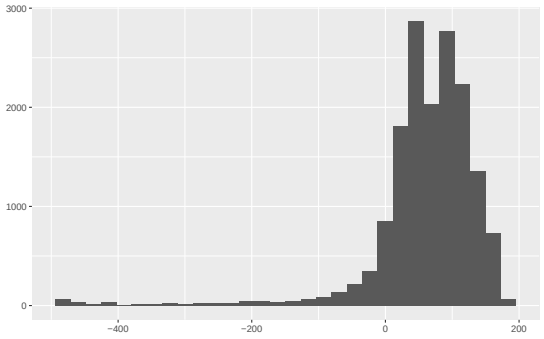
radial basis function with $\gamma \in \{0.001, 0.005, 0.01, 0.02\}$ against all combinations of $\nu \in \{0.07, 0.1, 0.15\}$. A selection of most promising results are shown in Figure 2.16. The results obtained with $\nu \in \{0.1, 0.15\}$ and $\gamma = 0.001$ shown in sub-figures a and c met the selection criterion the best.

The last iteration continued with the kernel set to the radial basis function with $\gamma \in \{0.0007, 0.002, 0.0055\}$ against all combinations of $\nu \in \{0.13, 0.18, 0.2\}$. A selection of most promising results are shown in Figure 2.17. The results obtained with $\nu \in \{0.18, 0.13\}$ and $\gamma = 0.002$ shown in sub-figures a and c best met the selection criterion and were chosen. They displayed a distinct set of outliers at the most negative end of the distribution. The top 100 outliers obtained by these two approaches were almost identical, with 98 being common.

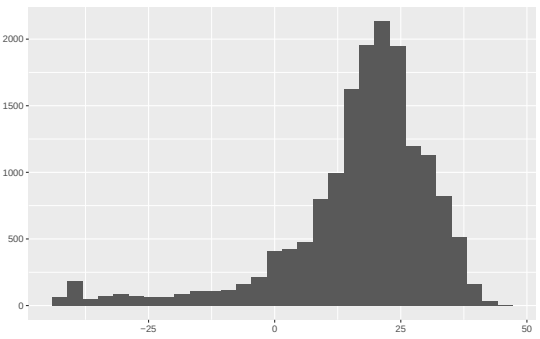
As was done with the K-nearest neighbour approach, the anomalies need to be considered in context. To move from contextual anomaly detection to point anomaly detection, the data set was broken up into subsets. The first subsets were based on node. Here the SVM decision values were calculated for each node separately. The hyper parameters first used were the same as those used in the last iteration above i.e. $\gamma \in \{0.0007, 0.002, 0.0055\}$ against all combinations of $\nu \in \{0.13, 0.18, 0.2\}$. The histograms of the SVM decision values for each combination is shown in Figure 2.18. The y axes have been scaled with the square root function.

Having γ set to 0.0007 and ν set to 0.2 showed the largest separation of the outliers from the rest of the observations. Table 2.15 showed that this combination of γ and ν also resulted it having 75 of its top 100 outliers in common with the results obtained with the analysis done on the group as a whole, using ν =0.13 and $\gamma = 0.002$.

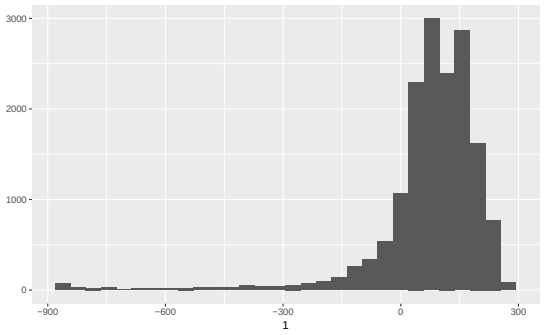
Further hyper parameter values were checked, building on those that provided the most promising results. The values what were checked were $\gamma \in \{0.0001, 0.0002, 0.0003\}$ against all combinations of $\nu \in \{0.3, 0.4, 0.5\}$. The histograms of the OCSVM decision values for each combination is shown in Figure 2.19. The y axes have been scaled with the square



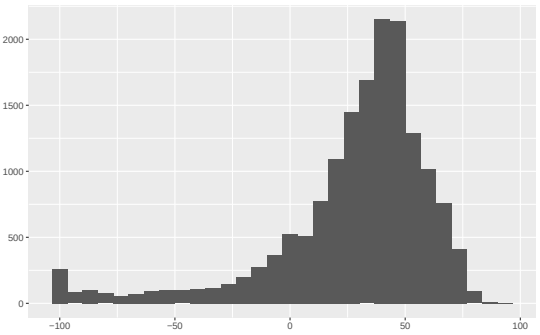
(a) $\nu = 0.1$ and $\gamma = 0.001$.



(b) $\nu = 0.1$ and $\gamma = 0.005$.

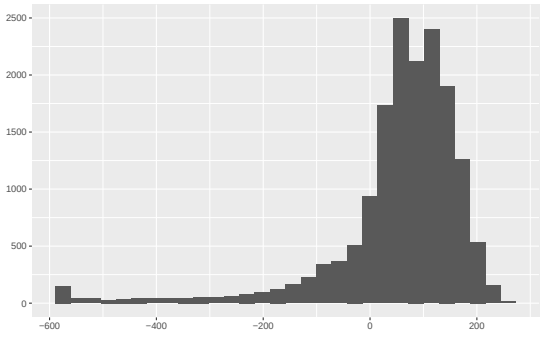


(c) $\nu = 0.15$ and γ =0.001.

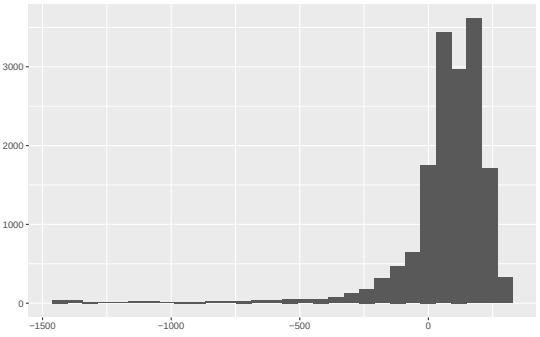


(d) $\nu = 0.15$ and γ =0.005.

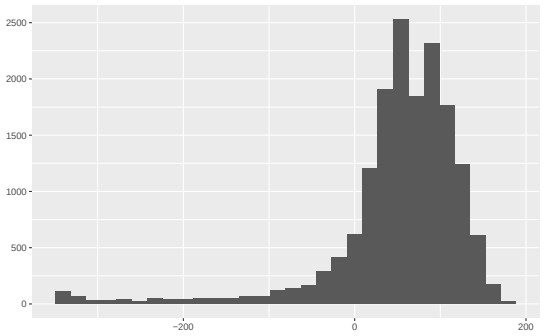
Figure 2.16: Resultant distributions in second hyper parameter search.



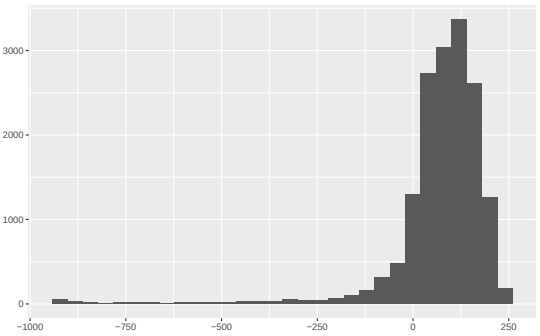
(a) $\nu = 0.18$ and $\gamma = 0.002$



(b) $\nu = 0.18$ and $\gamma = 0.007$

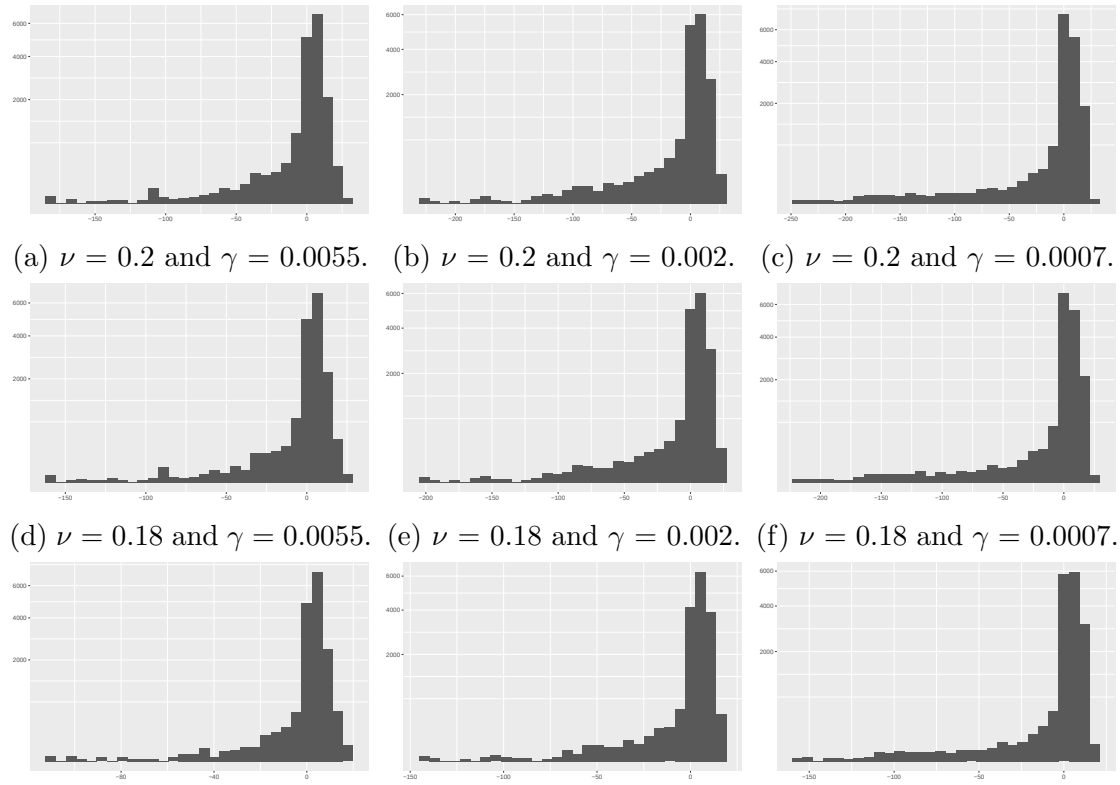


(c) $\nu = 0.13$ and γ =0.002



(d) $\nu = 0.13$ and γ =0.007

Figure 2.17: Resultant distributions in third hyper parameter search.



(a) $\nu = 0.2$ and $\gamma = 0.0055$. (b) $\nu = 0.2$ and $\gamma = 0.002$. (c) $\nu = 0.2$ and $\gamma = 0.0007$.

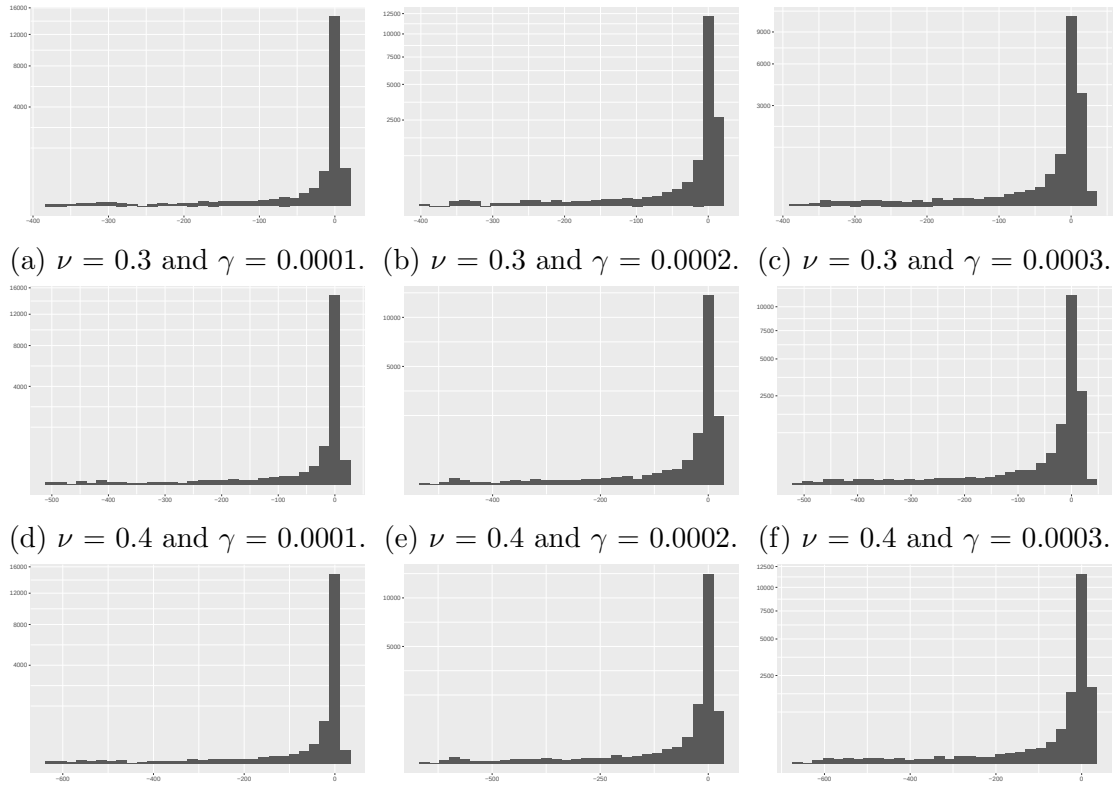
(d) $\nu = 0.18$ and $\gamma = 0.0055$. (e) $\nu = 0.18$ and $\gamma = 0.002$. (f) $\nu = 0.18$ and $\gamma = 0.0007$.

(g) $\nu = 0.13$ and $\gamma = 0.0055$. (h) $\nu = 0.13$ and $\gamma = 0.002$. (i) $\nu = 0.13$ and $\gamma = 0.0007$.

Figure 2.18: Resultant distributions in first node based search.

Table 2.15: Count of top outliers common with original non-subset OCSVM results.

	Common to top 100	Common to top 200	Common to top 300
$\nu = 0.13$,$\gamma = 0.0007$	68	167	241
$\nu = 0.13$,$\gamma = 0.002$	46	120	164
$\nu = 0.13$,$\gamma = 0.0055$	9	38	108
$\nu = 0.18$,$\gamma = 0.0007$	73	174	248
$\nu = 0.18$,$\gamma = 0.002$	48	120	164
$\nu = 0.18$,$\gamma = 0.0055$	9	38	91
$\nu = 0.2$,$\gamma = 0.0007$	75	176	248
$\nu = 0.2$,$\gamma = 0.002$	44	119	163
$\nu = 0.2$,$\gamma = 0.0055$	9	38	83



(a) $\nu = 0.3$ and $\gamma = 0.0001$. (b) $\nu = 0.3$ and $\gamma = 0.0002$. (c) $\nu = 0.3$ and $\gamma = 0.0003$.

(d) $\nu = 0.4$ and $\gamma = 0.0001$. (e) $\nu = 0.4$ and $\gamma = 0.0002$. (f) $\nu = 0.4$ and $\gamma = 0.0003$.

(g) $\nu = 0.5$ and $\gamma = 0.0001$. (h) $\nu = 0.5$ and $\gamma = 0.0002$. (i) $\nu = 0.5$ and $\gamma = 0.0003$.

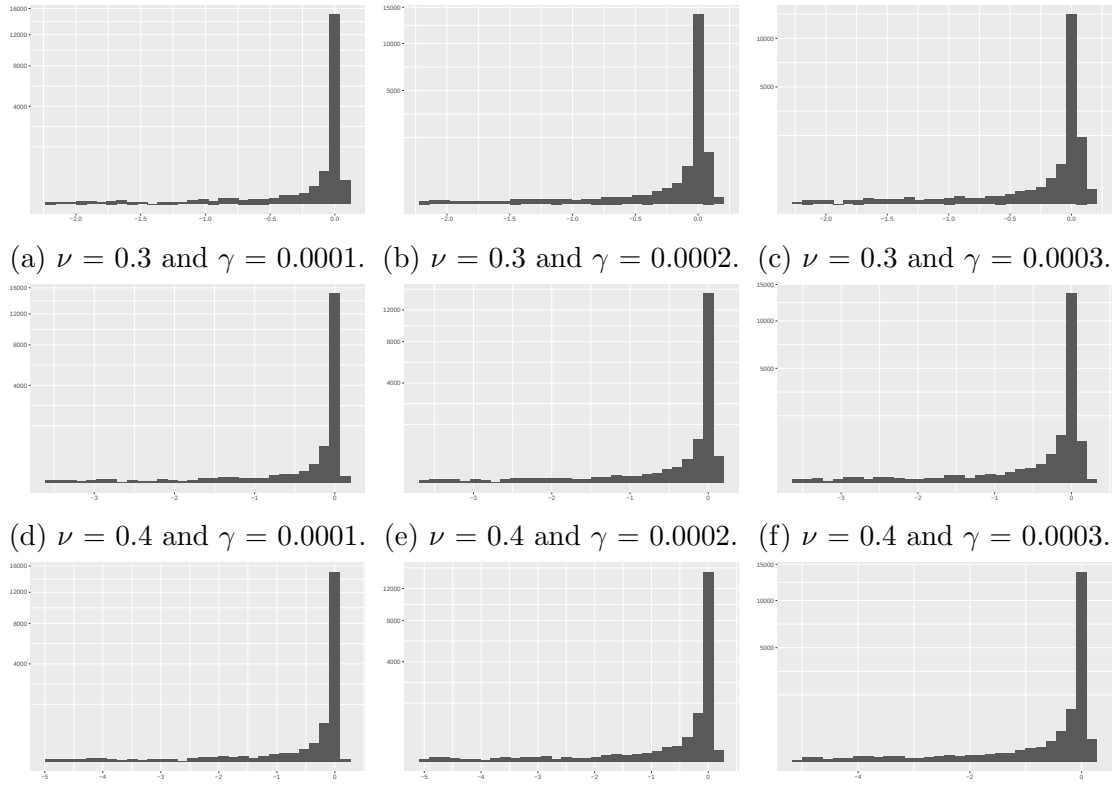
Figure 2.19: Resultant distributions in second node based search.

root function. This resulted in the outliers being separated out even further than the results shown in Figure 2.18.
These hyper parameters resulted in the top outliers identified being very similar to those identified by the OCSVM analysis done on the non-subset data with ν set to 0.13 and γ set to 0.002. This comparison is shown in Table 2.16.

Table 2.16: Second search resultant count of top outliers common with original non-subset OCSVM results.

	Common to top 100	Common to top 200	Common to top 300
$\nu = 0.3$,$\gamma = 0.0001$	92	189	271
$\nu = 0.3$,$\gamma = 0.0002$	91	189	270
$\nu = 0.3$,$\gamma = 0.0003$	90	186	265
$\nu = 0.4$,$\gamma = 0.0001$	92	190	274
$\nu = 0.4$,$\gamma = 0.0002$	92	189	272
$\nu = 0.4$,$\gamma = 0.0003$	91	187	268
$\nu = 0.5$,$\gamma = 0.0001$	92	190	277
$\nu = 0.5$,$\gamma = 0.0001$	92	190	274
$\nu = 0.5$,$\gamma = 0.0003$	91	187	272

As was done with the KNN approach the data was grouped into node/time based grouping. The histograms of the resultant decision values, with $\gamma \in \{0.0001, 0.0002, 0.0003\}$ against all combinations of $\nu \in \{0.3, 0.4, 0.5\}$ are shown in Figure 2.20. The distributions are similar those seen using the node based grouping. The anomalies identified by the node/time based approach are compared to those identified by the non-subset group and the node based grouping in Table 2.17 and Table 2.18 respectively. The comparison



(a) $\nu = 0.3$ and $\gamma = 0.0001$. (b) $\nu = 0.3$ and $\gamma = 0.0002$. (c) $\nu = 0.3$ and $\gamma = 0.0003$.

(d) $\nu = 0.4$ and $\gamma = 0.0001$. (e) $\nu = 0.4$ and $\gamma = 0.0002$. (f) $\nu = 0.4$ and $\gamma = 0.0003$.

(g) $\nu = 0.5$ and $\gamma = 0.0001$. (h) $\nu = 0.5$ and $\gamma = 0.0002$. (i) $\nu = 0.5$ and $\gamma = 0.0003$.

Figure 2.20: Resultant distributions in first node/time based search.

with the node based analysis had more outliers in common than the comparison with the non-subset analysis. Setting ν to 0.5 and γ to 0.0001 resulted in the highest number of common points in both cases. For comparison purposes the outliers identified by the node based and node/time based approaches obtained using ν set to 0.5 and γ set to 0.0001 will therefore be used. The outliers obtained in the non-subset approach using ν set to 0.13 and γ set to 0.002 will be used.

Table 2.17: First search resultant count of top outliers common with original non-subset OCSVM results.

	Common to top 100	Common to top 200	Common to top 300
$\nu = 0.3$,$\gamma = 0.0001$	90	179	258
$\nu = 0.3$,$\gamma = 0.0002$	87	175	254
$\nu = 0.3$,$\gamma = 0.0003$	84	170	248
$\nu = 0.4$,$\gamma = 0.0001$	90	184	264
$\nu = 0.4$,$\gamma = 0.0002$	89	181	264
$\nu = 0.4$,$\gamma = 0.0003$	88	178	262
$\nu = 0.5$,$\gamma = 0.0001$	91	185	269
$\nu = 0.5$,$\gamma = 0.0001$	90	183	266
$\nu = 0.5$,$\gamma = 0.0003$	90	182	265

2.3.4 Density-Based Local Outlier Factor (LOF)

The next approach used to identify outliers was introduced in [7]. The two main aims of this approach are to identify local outliers, that is, points that are outliers with respect to

Table 2.18: First search resultant count of top outliers common with node based subset OCSVM results.

	Common to top 100	Common to top 200	Common to top 300
$\nu = 0.3$,$\gamma = 0.0001$	92	185	269
$\nu = 0.3$,$\gamma = 0.0002$	90	185	266
$\nu = 0.3$,$\gamma = 0.0003$	88	180	260
$\nu = 0.4$,$\gamma = 0.0001$	92	188	269
$\nu = 0.4$,$\gamma = 0.0002$	92	188	267
$\nu = 0.4$,$\gamma = 0.0003$	91	187	267
$\nu = 0.5$,$\gamma = 0.0001$	91	187	272
$\nu = 0.5$,$\gamma = 0.0001$	91	188	271
$\nu = 0.5$,$\gamma = 0.0003$	92	190	271

a local neighbourhood and to assign a value (Local Outlier Factor) identifying how much of an outlier the points are. Figure 1.1 shows two clusters, the one on the right consists of points that are tightly packed together whereas the cluster on the left is less dense with the points being spread out. There are two outliers, one global outlier, marked with a "x" and one local outlier marked with a "o". Using the KNN neighbour approach for outlier detection, only the global outlier will be identified. There are no values for k that will enable KNN to detect the local outlier without incorrectly identifying observations in the sparse cluster to be outliers as well. The LOF approach aims to identify point "o" as well. The extent to which an observation is an outlier or not is given by its Local Outlier Factor (LOF). The closer a observation's LOF is to 1 the more normal the observation. The higher the LOF value the more of an outlier it is.
According to [7] the definition of LOF starts off with defining the distance to the kth nearest neighbour of observation p as $k\text{-}distance(p)$ and the neighbourhood of p as $N_{k\text{-}distance(p)}(p)$ which includes all the observations as close or closer to p than the k^th nearest neighbour. Equation 2.3 defines the reachability distance of point p. This is simply the greater of either the actual distance from o if the point p is further than the $k - distance(o)$ of o or if it is nearer, then it is defined as the $k - distance(o)$. This has the result of defining all the points in the neighbourhood of o as having a reachability distance of $k - distance(o)$ and those that are further as having a reachability distance of their actual distance. Doing this reduces the statistical fluctuations for all points close to o. These definitions together with the parameter $MinPts$ which defines the number of nearest neighbours used in defining the local neighborhood, are used to define the concept of the local reachability density of an object p as shown in Equation 2.2. Lastly Equation 2.1 defines the LOF of p.

$$LOF_{MinPts} = \frac{\sum_{o \in N_{MinPts}(p)} \frac{lrd_{MinPts}(o)}{lrd_{MinPts}(p)}}{|N_{MinPts}(p)|} \tag{2.1}$$

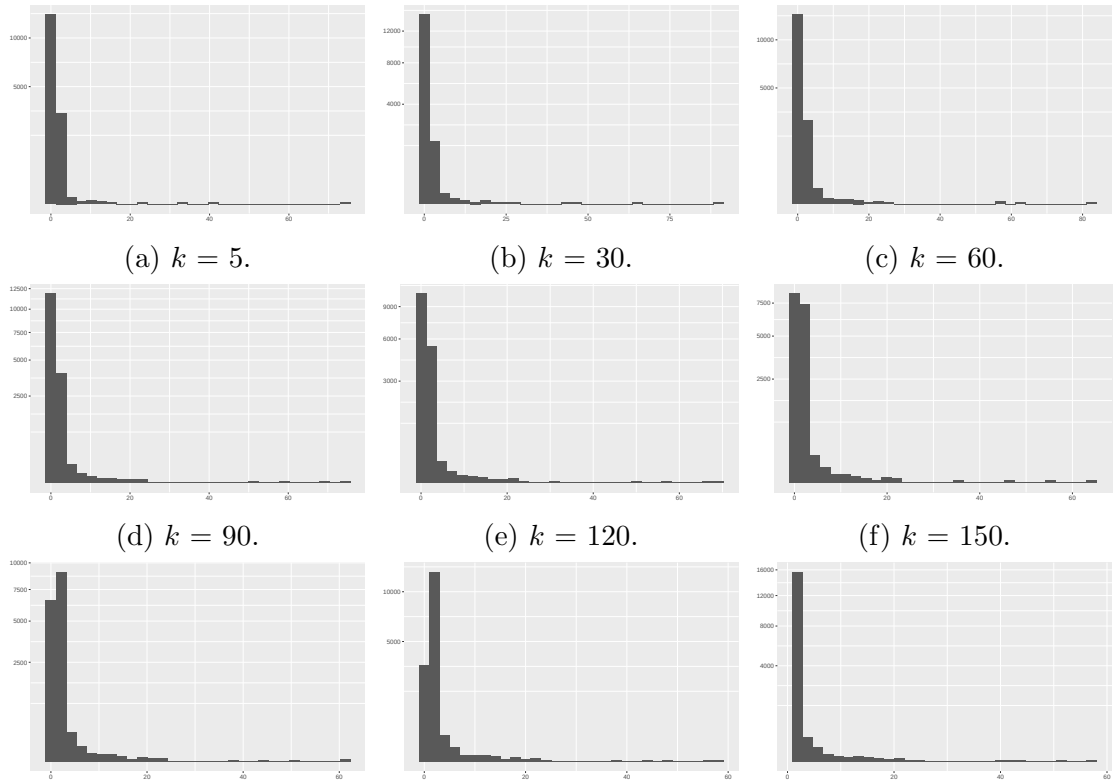
Where

$$lrd_{MinPts} = \frac{|N_{MinPts}(p)|}{\sum_{o \in N_{MinPts}(p)} reach\text{-}dist_{MinPts}(p, o)} \tag{2.2}$$

is the local reachability density of object p and

$$reach\text{-}dist_k = \max\{k\text{-}distance(o), d(p, o)\} \tag{2.3}$$

is the reachability distance of an object p from object o



(a) $k = 5$. (b) $k = 30$. (c) $k = 60$.

(d) $k = 90$. (e) $k = 120$. (f) $k = 150$.

(g) $k = 180$. (h) $k = 210$. (i) $k = 240$.

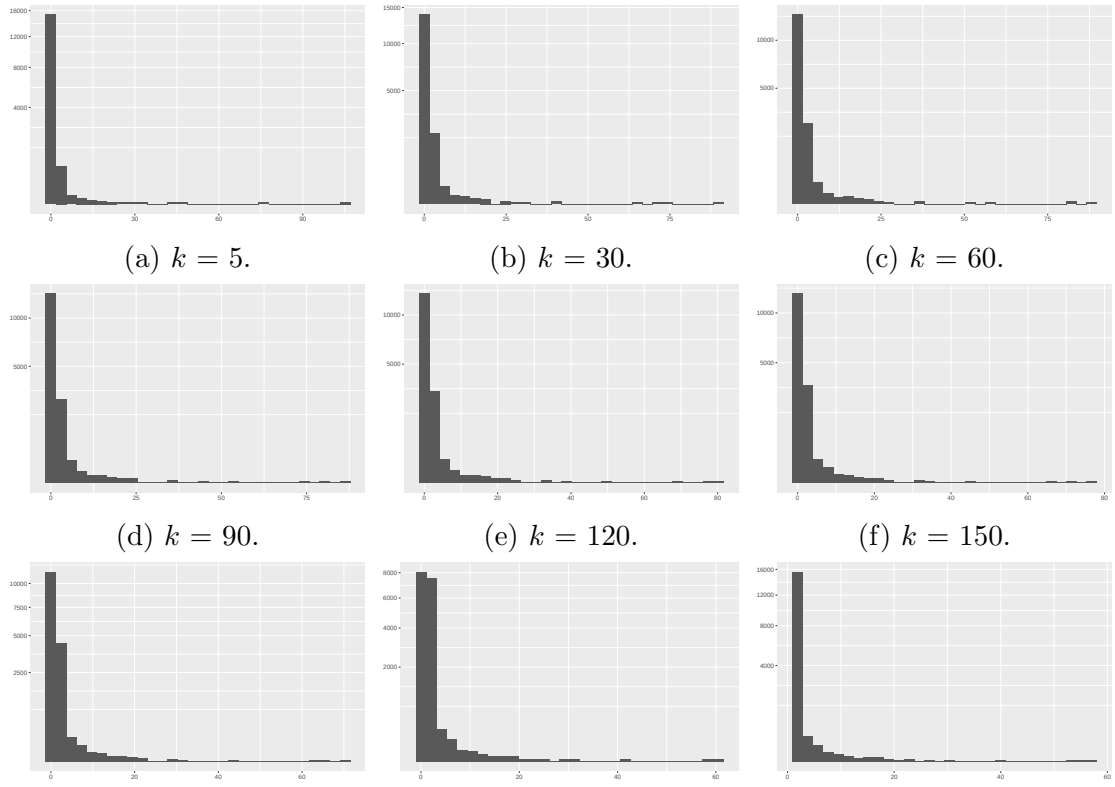
Figure 2.21: Resultant distributions of LOF values.

The lof function from the R package DBSCAN was used to create the LOF values for our SGW data set. The first approach was to treat the data set as a whole without any sub-setting. A range of k values were used, ranging from 5 to 250, in steps of 5. The distributions of 9 of the k values are shown in Figure 2.21. As the data set is in unlabeled, choice of k could not be based on accuracy of classification. The approach taken with the one class support vector machine for selecting the optimum hyper parameters was to choose those that resulted in distributions that separated outliers out furthest from the bulk of the observations. The distributions obtained for the various k values all gave similar results with no one value of k showing a more promising distribution. To further investigate the differences in the outliers identified by the different k values, the top 100 outliers for each k value was compared to the outliers identified when using the remaining k values. Table 2.19 shows these results. The lower the values, the darker the back ground shading, with 0 being black and 100 being white. Added to this all values greater or equal to 80 have a grayed out font. This highlights that for k values above 95, the outliers identified are very similar, with more than 80% being common. The outliers identified when using a k value of 100 were used for comparison with the other techniques used in the study.

As was done with the K-nearest neighbour and one-class support vector machine approaches, the anomalies also need to be considered in context. To move from contextual anomaly detection to point anomaly detection, the data set was be broken up into subsets based on node. Here the LOF values were calculated for each node separately. The same set of k values as used in the ungrouped approach were used i.e. ranging from 5 to 250, in steps of 5. As with the ungrouped approach, the distributions obtained gave no guidance in selecting the optimum value for k as they all have very similar distributions. A selection of these distributions are shown in Figure 2.22. It is clear from Table 2.20 that the outliers obtained by the different k values are less diverse than those obtained from the ungrouped

Table 2.19: Comparison of LOF results with differing k values, showing matching outlier count per pair.

k	5	10	15	20	25	30	35	40	45	50	55	60	65	70	75	80	85	90	95	100	105	110	115	120	125	130	135	140	145	150	155	160	165	170	175	180	185	190	195	200	205	210	215	220	225	230	235	240	245	250
5	100	71	54	46	41	38	37	36	33	31	31	31	31	31	30	29	29	29	27	27	27	27	27	27	28	28	28	28	27	26	26	26	26	26	26	25	25	25	25	24	24	24	24	24	24	23	23	23	23	23
10	71	100	74	65	59	56	55	54	51	49	49	46	44	43	41	40	39	39	38	37	37	37	37	37	38	37	37	37	35	34	34	34	34	34	34	33	33	33	33	32	33	33	33	33	33	33	33	33	33	33
15	54	74	100	89	83	79	78	77	74	72	71	68	64	62	58	57	53	50	49	48	48	48	48	48	49	48	48	47	46	45	45	45	45	45	45	44	44	44	43	42	43	43	43	43	43	43	43	42	42	42
20	46	65	89	100	94	90	87	86	83	81	80	76	72	69	65	64	60	57	56	55	55	55	55	55	56	55	55	54	53	52	52	52	52	52	52	51	51	51	50	49	50	50	50	50	50	50	50	49	49	49
25	41	59	83	94	100	95	92	91	88	86	85	81	77	74	70	69	65	62	59	59	59	59	59	59	60	59	58	57	56	55	55	55	55	55	55	54	54	54	53	52	53	53	53	53	53	53	53	52	52	52
30	38	56	79	90	95	100	97	96	93	91	90	86	82	79	75	73	69	66	63	63	63	63	63	63	64	63	62	61	60	59	59	59	59	59	59	58	58	58	57	56	57	57	57	57	57	57	57	56	56	56
35	37	55	78	87	92	97	100	99	96	94	93	89	85	82	78	76	72	69	66	66	66	66	66	66	67	66	65	64	63	62	62	61	61	61	61	60	60	60	59	58	59	59	59	59	59	59	59	58	58	58
40	36	54	77	86	91	96	99	100	97	95	94	90	86	83	79	77	73	70	67	67	67	67	67	67	67	66	65	64	63	62	62	61	61	61	61	60	60	60	59	58	59	59	59	59	59	59	59	58	58	58
45	33	51	74	83	88	93	96	97	100	98	97	93	89	86	82	79	75	72	69	69	69	69	69	69	69	68	67	66	65	64	64	63	63	63	63	62	62	62	61	60	61	61	61	61	61	61	61	60	60	60
50	31	49	72	81	86	91	94	95	98	100	99	95	91	88	84	81	77	74	71	71	71	71	71	71	71	70	69	68	67	66	66	65	65	65	65	64	64	64	63	62	63	63	63	63	63	63	63	62	62	62
55	31	49	71	80	85	90	93	94	97	99	100	96	92	89	85	82	78	75	72	72	72	72	72	72	72	71	70	69	68	67	67	66	66	66	66	65	65	65	64	63	64	64	64	64	64	64	64	63	63	63
60	31	46	68	76	81	86	89	90	93	95	96	100	96	93	89	86	82	79	76	76	75	75	75	75	74	73	72	71	70	69	68	68	68	68	68	67	67	67	66	65	64	64	64	65	65	64	63	62	62	62
65	31	44	64	72	77	82	85	86	89	91	92	96	100	97	93	90	86	83	80	80	79	79	79	79	78	77	76	75	74	73	72	72	72	72	72	72	71	71	71	70	69	68	68	68	69	69	68	67	66	66
70	31	43	62	69	74	79	82	83	86	88	89	93	97	100	96	93	89	86	83	83	82	81	81	81	80	79	77	76	75	74	73	73	73	73	73	72	72	72	71	70	69	69	69	70	70	69	68	67	67	67
75	30	41	58	65	70	75	78	79	82	84	85	89	93	96	100	97	93	90	87	87	86	85	85	85	84	83	81	80	78	77	76	76	76	76	76	75	75	75	74	73	72	72	72	73	73	72	71	70	70	70
80	29	40	57	64	69	73	76	77	79	81	82	86	90	93	97	100	96	93	90	90	89	88	88	88	87	86	84	83	81	80	79	79	79	79	79	78	78	78	77	76	75	75	75	76	76	75	74	73	73	73
85	29	39	53	60	65	69	72	73	75	77	78	82	86	89	93	96	100	97	94	94	93	92	92	92	91	90	88	87	85	84	83	83	83	83	83	82	82	82	81	80	79	79	79	80	80	79	78	77	77	77
90	29	39	50	57	62	66	69	70	72	74	75	79	83	86	90	93	97	100	97	97	96	95	95	95	94	93	91	90	88	87	86	86	86	86	86	85	85	84	83	82	81	81	81	82	82	81	80	79	79	79
95	27	38	49	56	59	63	66	67	69	71	72	76	80	83	87	90	94	97	100	99	98	97	97	97	96	95	93	92	90	89	88	88	88	88	88	87	87	86	85	84	83	83	83	83	83	82	81	80	80	80
100	27	37	48	55	59	63	66	67	69	71	72	76	80	83	87	90	94	97	99	100	99	98	98	98	97	96	94	93	91	90	89	89	89	89	89	88	88	87	86	85	84	84	84	84	84	83	82	81	81	81
105	27	37	48	55	59	63	66	67	69	71	72	75	79	82	86	89	93	96	98	99	100	99	99	99	98	97	95	94	92	91	90	90	90	90	90	89	89	88	87	86	85	85	85	85	85	84	83	82	82	82
110	27	37	48	55	59	63	66	67	69	71	72	75	79	81	85	88	92	95	97	98	99	100	99	99	98	97	95	95	93	92	91	91	91	91	91	90	90	89	88	87	86	86	86	86	86	86	85	84	83	83
115	27	37	48	55	59	63	66	67	69	71	72	75	79	81	85	88	92	95	97	98	99	99	100	100	99	98	96	95	93	92	91	91	91	91	91	90	90	89	88	87	86	86	86	86	86	86	85	84	83	83
120	27	37	48	55	59	63	66	67	69	71	72	75	79	81	85	88	92	95	97	98	99	99	100	100	99	98	96	95	93	92	91	91	91	91	91	90	90	89	88	87	86	86	86	86	86	86	85	84	83	83
125	28	38	49	56	60	64	67	67	69	71	72	74	78	80	84	87	91	94	96	97	98	98	99	99	100	99	97	96	94	93	92	92	92	92	92	91	91	90	89	88	87	87	87	87	87	87	86	85	84	84
130	28	37	48	55	59	63	66	66	68	70	71	73	77	79	83	86	90	93	95	96	97	97	98	98	99	100	98	97	95	94	93	93	93	93	93	92	92	91	90	89	88	88	88	88	88	88	87	86	85	85
135	28	37	48	55	58	62	65	65	67	69	70	72	76	77	81	84	88	91	93	94	95	95	96	96	97	98	100	99	97	96	95	95	95	95	95	94	94	93	92	91	90	90	90	90	90	89	88	87	87	87
140	28	37	47	54	57	61	64	64	66	68	69	71	75	76	80	83	87	90	92	93	94	95	95	95	96	97	99	100	98	97	96	96	96	96	96	95	95	94	93	92	91	91	91	91	91	90	89	88	88	88
145	27	35	46	53	56	60	63	63	65	67	68	70	74	75	78	81	85	88	90	91	92	93	93	93	94	95	97	98	100	99	98	98	98	98	98	97	97	96	95	94	93	93	93	93	93	92	91	90	90	90
150	26	34	45	52	55	59	62	62	64	66	67	69	73	74	77	80	84	87	89	90	91	92	92	92	93	94	96	97	99	100	99	99	99	99	99	98	98	97	96	95	94	94	94	94	94	93	92	91	91	91
155	26	34	45	52	55	59	62	62	64	66	67	68	72	73	76	79	83	86	88	89	90	91	91	91	92	93	95	96	98	99	100	99	99	99	99	98	98	97	96	95	95	95	95	94	94	94	93	93	92	92
160	26	34	45	52	55	59	61	61	63	65	66	68	72	73	76	79	83	86	88	89	90	91	91	91	92	93	95	96	98	99	99	100	100	100	100	99	99	98	97	96	95	95	95	95	95	95	94	93	92	92
165	26	34	45	52	55	59	61	61	63	65	66	68	72	73	76	79	83	86	88	89	90	91	91	91	92	93	95	96	98	99	99	100	100	100	100	99	99	98	97	96	95	95	95	95	95	95	94	93	92	92
170	26	34	45	52	55	59	61	61	63	65	66	68	72	73	76	79	83	86	88	89	90	91	91	91	92	93	95	96	98	99	99	100	100	100	100	99	99	98	97	96	95	95	95	95	95	95	94	93	92	92
175	26	34	45	52	55	59	61	61	63	65	66	68	72	73	76	79	83	86	88	89	90	91	91	91	92	93	95	96	98	99	99	100	100	100	100	99	99	98	97	96	95	95	95	95	95	95	94	93	92	92
180	25	33	44	51	54	58	60	60	62	64	65	67	71	72	75	78	82	85	87	88	89	90	90	90	91	92	94	95	97	98	98	99	99	99	99	100	100	99	98	97	96	96	96	96	96	95	94	93	93	93
185	25	33	44	51	54	58	60	60	62	64	65	67	71	72	75	78	82	85	87	88	89	90	90	90	91	92	94	95	97	98	98	99	99	99	99	100	100	99	98	97	96	96	96	96	96	95	94	93	93	93
190	25	33	44	51	54	58	60	60	62	64	65	67	71	72	75	78	82	84	86	87	88	89	89	89	90	91	93	94	96	97	97	98	98	98	98	99	99	100	99	98	97	97	97	97	97	96	95	94	94	94
195	25	33	43	50	53	57	59	59	61	63	64	66	70	71	74	77	81	83	85	86	87	88	88	88	89	90	92	93	95	96	96	97	97	97	97	98	98	99	100	99	98	98	98	98	98	97	96	95	95	95
200	24	32	42	49	52	56	58	58	60	62	63	65	69	70	73	76	80	82	84	85	86	87	87	87	88	89	91	92	94	95	95	96	96	96	96	97	97	98	99	100	99	99	99	99	99	98	97	96	96	96
205	24	33	43	50	53	57	59	59	61	63	64	64	68	69	72	75	79	81	83	84	85	86	86	86	87	88	90	91	93	94	95	95	95	95	95	96	96	97	98	99	100	100	100	99	99	98	98	97	97	97
210	24	33	43	50	53	57	59	59	61	63	64	64	68	69	72	75	79	81	83	84	85	86	86	86	87	88	90	91	93	94	95	95	95	95	95	96	96	97	98	99	100	100	100	99	99	98	98	97	97	97
215	24	33	43	50	53	57	59	59	61	63	64	64	68	69	72	75	79	81	83	84	85	86	86	86	87	88	90	91	93	94	95	95	95	95	95	96	96	97	98	99	100	100	100	99	99	98	98	97	97	97
220	24	33	43	50	53	57	59	59	61	63	64	65	69	70	73	76	80	82	83	84	85	86	86	86	87	88	90	91	93	94	94	95	95	95	95	96	96	97	98	99	99	99	99	100	100	99	98	97	97	97
225	24	33	43	50	53	57	59	59	61	63	64	65	69	70	73	76	80	82	83	84	85	86	86	86	87	88	90	91	93	94	94	95	95	95	95	96	96	97	98	99	99	99	99	100	100	99	98	97	97	97
230	23	33	43	50	53	57	59	59	61	63	64	64	68	69	72	75	79	81	82	83	84	85	85	85	86	87	89	90	92	93	93	94	94	94	94	95	95	96	97	98	98	98	98	99	99	100	99	98	98	98
235	23	33	43	50	53	57	59	59	61	63	64	63	67	68	71	74	78	80	81	82	83	84	84	84	85	86	88	89	91	92	93	93	93	93	93	94	94	95	96	97	98	98	98	98	98	99	100	99	99	99
240	23	33	42	49	52	56	58	58	60	62	63	62	66	67	70	73	77	79	80	81	82	83	83	83	84	85	87	88	90	91	92	92	92	92	92	93	93	94	95	96	97	97	97	97	97	98	99	100	100	100
245	23	33	42	49	52	56	58	58	60	62	63	62	66	67	70	73	77	79	80	81	82	83	83	83	84	85	87	88	90	91	92	92	92	92	92	93	93	94	95	96	97	97	97	97	97	98	99	100	100	100
250	23	33	42	49	52	56	58	58	60	62	63	62	66	67	70	73	77	79	80	81	82	83	83	83	84	85	87	88	90	91	92	92	92	92	92	93	93	94	95	96	97	97	97	97	97	98	99	100	100	100



(a) $k = 5$. (b) $k = 30$. (c) $k = 60$.

(d) $k = 90$. (e) $k = 120$. (f) $k = 150$.

(g) $k = 180$. (h) $k = 210$. (i) $k = 240$.

Figure 2.22: Resultant distributions of LOF after grouping by node.

data set. Here the biggest difference is between the outlier identified with k being 5 and with k being 110. These had 44 outliers in common. For all k values above 40, there are more than 80% of the outliers identified in common. The outliers identified when using a k value of 100 were used for comparison with the other techniques used in the study.
The node/time based approach was not attempted with the LOF based approach as there were only 12 observation in each subset and the minimum recommended k value to be used with the LOF approach is 10.

2.3.5 Multivariate Gaussian Distribution

The last approach uses a parametric model, the multivariate Gaussian distribution to identify outliers in the SGW dataset. The multivariate Gaussian distribution is described by the probability density function defined in Equation 2.4. Here the mean of each variable is captured the in vector $\boldsymbol{\mu}$ and k represents the number of variables. $\boldsymbol{\Sigma}$ defines the variance covariance matrix of the observations which captures correlations between features.

$$p(\boldsymbol{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) = (2\pi)^{\frac{-k}{2}} |\boldsymbol{\Sigma}|^{\frac{-1}{2}} e^{-1/2(\boldsymbol{x}-\boldsymbol{\mu})' \boldsymbol{\Sigma}^{-1} (\boldsymbol{x}-\boldsymbol{\mu})} \tag{2.4}$$

The calculation of $\boldsymbol{\Sigma}$ requires that the observation count be greater than number of features and that there be no duplicate features, otherwise $\boldsymbol{\Sigma}$ is not invertible and Equation 2.4 can not be calculated. Once the the probability density function has been calculated, the probability of each observation can be calculated for each point in the data set. The observations with the lowest probabilities are flagged as outliers.
As with the other approaches above the dataset was first analyzed as a whole and then divided up into a dataset for each node and each subset analyzed individually. Of the top 100 outliers identified by these two approaches only 42 were common. The node/time

Table 2.20: Comparison of LOF results based on node grouping with differing k values, showing matching outlier count per pair.

k	5	10	15	20	25	30	35	40	45	50	55	60	65	70	75	80	85	90	95	100	105	110	115	120	125	130	135	140	145	150	155	160	165	170	175	180	185	190	195	200	205	210	215	220	225	230	235	240	245	250
5	100	80	74	66	58	53	51	49	49	49	47	47	46	44	44	45	45	45	45	45	45	44	44	45	45	45	45	45	45	45	45	45	45	45	45	45	45	45	46	47	48	48	48	48	48	48	47	47	47	47
10	80	100	93	82	73	68	65	63	63	63	62	62	61	59	59	60	60	60	60	60	60	59	59	59	59	59	59	58	57	58	57	57	57	57	56	56	56	56	57	57	58	58	58	58	58	59	59	59	59	59
15	74	93	100	89	79	74	71	69	68	68	67	68	67	66	66	66	66	66	66	66	66	65	65	65	65	65	65	64	63	64	63	63	63	63	62	62	61	60	61	61	62	62	62	62	62	63	63	63	63	63
20	66	82	89	100	90	83	80	78	77	77	75	76	75	74	73	73	73	73	73	73	73	72	72	72	72	72	72	71	70	71	70	70	70	70	69	69	68	67	68	68	69	69	69	69	69	70	70	70	70	70
25	58	73	79	90	100	93	90	88	87	87	84	84	83	82	81	81	80	80	80	80	80	79	79	79	79	79	79	78	77	78	77	77	77	77	76	76	76	75	75	75	75	75	75	75	75	75	76	76	76	76
30	53	68	74	83	93	100	96	95	94	94	91	90	89	88	87	86	85	85	84	84	84	83	83	83	83	83	83	82	81	82	81	81	81	81	80	80	80	79	79	79	78	78	78	78	78	79	79	79	79	79
35	51	65	71	80	90	96	100	98	97	97	93	92	91	90	89	88	87	87	86	86	85	84	84	84	84	84	84	83	82	83	82	82	82	82	81	81	81	80	80	80	79	79	79	79	79	80	80	80	80	80
40	49	63	69	78	88	95	98	100	99	99	95	94	93	92	91	89	89	89	88	88	87	86	86	85	85	85	85	84	83	84	83	83	83	83	82	82	82	81	81	81	80	80	80	80	80	81	81	81	81	81
45	49	63	68	77	87	94	97	99	100	100	96	95	94	92	91	90	90	90	89	89	88	87	87	86	86	86	86	85	84	85	84	84	84	84	83	83	83	82	82	82	81	81	81	81	81	82	82	82	82	82
50	49	63	68	77	87	94	97	99	100	100	96	95	94	92	91	90	90	90	89	89	88	87	87	86	86	86	86	85	84	85	84	84	84	84	83	83	83	82	82	82	81	81	81	81	81	82	82	82	82	82
55	47	62	67	75	84	91	93	95	96	96	100	99	98	96	95	94	94	94	93	93	92	91	91	90	90	90	90	89	88	89	88	88	88	88	87	87	87	86	86	85	84	84	84	84	84	84	84	84	84	84
60	47	62	68	76	84	90	92	94	95	95	99	100	99	97	96	95	95	95	94	94	93	92	92	91	91	91	91	90	89	90	89	89	89	89	88	88	87	86	86	85	84	84	84	84	84	84	84	84	84	84
65	46	61	67	75	83	89	91	93	94	94	98	99	100	98	97	96	96	96	95	95	94	93	93	92	92	92	92	91	90	91	90	90	90	90	89	89	88	87	87	86	85	85	85	85	85	85	85	85	85	85
70	44	59	66	74	82	88	90	92	92	92	96	97	98	100	99	97	97	97	96	96	95	94	94	93	93	93	93	91	90	91	90	90	90	90	89	89	88	87	87	86	85	85	85	85	85	85	85	85	85	85
75	44	59	66	73	81	87	89	91	91	91	95	96	97	99	100	98	98	98	97	97	96	95	95	94	94	94	94	92	91	92	91	91	91	91	90	90	89	88	88	87	86	86	86	86	86	86	86	86	86	86
80	45	60	66	73	81	86	88	89	90	90	94	95	96	97	98	100	99	99	98	98	97	96	96	96	96	96	96	94	93	94	93	93	93	93	92	92	91	90	89	88	87	87	87	87	87	87	87	87	87	87
85	45	60	66	73	80	85	87	89	90	90	94	95	96	97	98	99	100	100	99	99	98	97	97	96	96	96	96	94	93	94	93	93	93	93	92	92	91	90	89	88	87	87	87	87	87	87	87	87	87	87
90	45	60	66	73	80	85	87	89	90	90	94	95	96	97	98	99	100	100	99	99	98	97	97	96	96	96	96	94	93	94	93	93	93	93	92	92	91	90	89	88	87	87	87	87	87	87	87	87	87	87
95	45	60	66	73	80	84	86	88	89	89	93	94	95	96	97	98	99	99	100	100	99	98	98	97	97	97	97	95	94	95	94	94	94	94	93	93	92	91	90	89	88	88	88	88	88	88	88	88	88	88
100	45	60	66	73	80	84	86	88	89	89	93	94	95	96	97	98	99	99	100	100	99	98	98	97	97	97	97	95	94	95	94	94	94	94	93	93	92	91	90	89	88	88	88	88	88	88	88	88	88	88
105	45	60	66	73	80	84	85	87	88	88	92	93	94	95	96	97	98	98	99	99	100	99	99	98	98	98	98	96	95	96	95	95	95	95	94	94	93	92	91	90	89	89	89	89	89	89	89	89	89	89
110	44	59	65	72	79	83	84	86	87	87	91	92	93	94	95	96	97	97	98	98	99	100	100	99	99	99	99	97	96	97	96	96	96	96	95	95	94	93	92	91	90	90	90	90	90	90	90	90	90	90
115	44	59	65	72	79	83	84	86	87	87	91	92	93	94	95	96	97	97	98	98	99	100	100	99	99	99	99	97	96	97	96	96	96	96	95	95	94	93	92	91	90	90	90	90	90	90	90	90	90	90
120	45	59	65	72	79	83	84	85	86	86	90	91	92	93	94	96	96	96	97	97	98	99	99	100	100	100	100	98	97	98	97	97	97	97	96	96	95	94	93	92	91	91	91	91	91	90	90	90	90	90
125	45	59	65	72	79	83	84	85	86	86	90	91	92	93	94	96	96	96	97	97	98	99	99	100	100	100	100	98	97	98	97	97	97	97	96	96	95	94	93	92	91	91	91	91	91	90	90	90	90	90
130	45	59	65	72	79	83	84	85	86	86	90	91	92	93	94	96	96	96	97	97	98	99	99	100	100	100	100	98	97	98	97	97	97	97	96	96	95	94	93	92	91	91	91	91	91	90	90	90	90	90
135	45	59	65	72	79	83	84	85	86	86	90	91	92	93	94	96	96	96	97	97	98	99	99	100	100	100	100	98	97	98	97	97	97	97	96	96	95	94	93	92	91	91	91	91	91	90	90	90	90	90
140	45	58	64	71	78	82	83	84	85	85	89	90	91	91	92	94	94	94	95	95	96	97	97	98	98	98	98	100	99	99	98	98	98	98	97	97	96	95	94	93	92	92	92	92	92	91	91	91	91	91
145	45	57	63	70	77	81	82	83	84	84	88	89	90	90	91	93	93	93	94	94	95	96	96	97	97	97	97	99	100	99	99	99	99	99	98	98	97	96	95	94	93	93	93	93	93	92	92	92	92	92
150	45	58	64	71	78	82	83	84	85	85	89	90	91	91	92	94	94	94	95	95	96	97	97	98	98	98	98	99	99	100	98	98	98	98	97	97	96	95	94	93	92	92	92	92	92	91	91	91	91	91
155	45	57	63	70	77	81	82	83	84	84	88	89	90	90	91	93	93	93	94	94	95	96	96	97	97	97	97	98	99	98	100	100	100	100	99	99	98	97	96	95	94	94	94	94	94	93	93	93	93	93
160	45	57	63	70	77	81	82	83	84	84	88	89	90	90	91	93	93	93	94	94	95	96	96	97	97	97	97	98	99	98	100	100	100	100	99	99	98	97	96	95	94	94	94	94	94	93	93	93	93	93
165	45	57	63	70	77	81	82	83	84	84	88	89	90	90	91	93	93	93	94	94	95	96	96	97	97	97	97	98	99	98	100	100	100	100	99	99	98	97	96	95	94	94	94	94	94	93	93	93	93	93
170	45	57	63	70	77	81	82	83	84	84	88	89	90	90	91	93	93	93	94	94	95	96	96	97	97	97	97	98	99	98	100	100	100	100	99	99	98	97	96	95	94	94	94	94	94	93	93	93	93	93
175	45	56	62	69	76	80	81	82	83	83	87	88	89	89	90	92	92	92	93	93	94	95	95	96	96	96	96	97	98	97	99	99	99	99	100	100	99	98	97	96	95	95	95	95	95	94	93	93	93	93
180	45	56	62	69	76	80	81	82	83	83	87	88	89	89	90	92	92	92	93	93	94	95	95	96	96	96	96	97	98	97	99	99	99	99	100	100	99	98	97	96	95	95	95	95	95	94	93	93	93	93
185	45	56	61	68	76	80	81	82	83	83	87	87	88	88	89	91	91	91	92	92	93	94	94	95	95	95	95	96	97	96	98	98	98	98	99	99	100	99	98	97	96	96	96	96	96	95	94	94	94	94
190	45	56	60	67	75	79	80	81	82	82	86	86	87	87	88	90	90	90	91	91	92	93	93	94	94	94	94	95	96	95	97	97	97	97	98	98	99	100	99	98	97	97	97	97	97	96	95	95	95	95
195	46	57	61	68	75	79	80	81	82	82	86	86	87	87	88	89	89	89	90	90	91	92	92	93	93	93	93	94	95	94	96	96	96	96	97	97	98	99	100	99	98	98	98	98	98	97	96	96	96	96
200	47	57	61	68	75	79	80	81	82	82	85	85	86	86	87	88	88	88	89	89	90	91	91	92	92	92	92	93	94	93	95	95	95	95	96	96	97	98	99	100	99	99	99	99	99	98	97	97	97	97
205	48	58	62	69	75	78	79	80	81	81	84	84	85	85	86	87	87	87	88	88	89	90	90	91	91	91	91	92	93	92	94	94	94	94	95	95	96	97	98	99	100	100	100	100	100	99	98	98	98	98
210	48	58	62	69	75	78	79	80	81	81	84	84	85	85	86	87	87	87	88	88	89	90	90	91	91	91	91	92	93	92	94	94	94	94	95	95	96	97	98	99	100	100	100	100	100	99	98	98	98	98
215	48	58	62	69	75	78	79	80	81	81	84	84	85	85	86	87	87	87	88	88	89	90	90	91	91	91	91	92	93	92	94	94	94	94	95	95	96	97	98	99	100	100	100	100	100	99	98	98	98	98
220	48	58	62	69	75	78	79	80	81	81	84	84	85	85	86	87	87	87	88	88	89	90	90	91	91	91	91	92	93	92	94	94	94	94	95	95	96	97	98	99	100	100	100	100	100	99	98	98	98	98
225	48	58	62	69	75	78	79	80	81	81	84	84	85	85	86	87	87	87	88	88	89	90	90	91	91	91	91	92	93	92	94	94	94	94	95	95	96	97	98	99	100	100	100	100	100	99	98	98	98	98
230	48	59	63	70	76	79	80	81	82	82	84	84	85	85	86	87	87	87	88	88	89	90	90	90	90	90	90	91	92	91	93	93	93	93	93	94	94	95	96	97	98	99	99	99	99	99	100	99	99	99
235	47	59	63	70	76	79	80	81	82	82	84	84	85	85	86	87	87	87	88	88	89	90	90	90	90	90	90	91	92	91	93	93	93	93	93	93	93	94	95	96	97	98	98	98	98	98	99	100	100	100
240	47	59	63	70	76	79	80	81	82	82	84	84	85	85	86	87	87	87	88	88	89	90	90	90	90	90	90	91	92	91	93	93	93	93	93	93	94	95	96	97	98	98	98	98	98	99	100	100	100	100
245	47	59	63	70	76	79	80	81	82	82	84	84	85	85	86	87	87	87	88	88	89	90	90	90	90	90	90	91	92	91	93	93	93	93	93	93	94	95	96	97	98	98	98	98	98	99	100	100	100	100
250	47	59	63	70	76	79	80	81	82	82	84	84	85	85	86	87	87	87	88	88	89	90	90	90	90	90	90	91	92	91	93	93	93	93	93	93	94	95	96	97	98	98	98	98	98	99	100	100	100	100

subsets were not analyzed as this grouping resulted in 13 observations per group. As the number of variables are greater than the number of observations, $\mathbf{\Sigma}$ is not invertible and the probability density function could not be calculated.

2.4 Comparison of the Results of the Unsupervised Approaches

Table 2.21 compares the top 100 outliers identified by each of the 10 different approaches described above. This is a pair wise comparison showing each approach compared to the other 9 approaches, showing how many of the top 100 outliers identified by the pair are common. Overall the different approaches ended up identifying many of the same points as outliers. The two approaches with the most similar top 100 results were KNN node/time and OCSVM node where 98 were common. The two most dissimilar approaches were the MGD node and OCSVM node/time approaches where only 34 were common. The MGD node approach in general had the least top 100 outliers in common with any of the other approaches. The approach that it has most in common with was the LOF node approach. This was even higher than it had in common with the MGD approach where only 42 were in common. When comparing the total, node and node/time results within each of the other unsupervised approaches the LOF based approach comes out with the next least similar results with the LOF and LOF node approaches having 74 in common. This difference could point to the dataset containing global anomalies as [4] finds that LOF performs poorly when this is the case. Added to this [4] concludes that global anomaly detection algorithms result in average results when used on datasets with local anomalies. The OCSVM and OC SVM node approaches had 92 to 100 observation in common. This was the same as the KNN and KNN node/time approaches which also had 92 observations in common.

2.5 Creating an Anomaly Free Data Set

This section describes how the outlier information generated by the 10 different outlier detection approaches described in the previous section are used to remove outliers from the original raw dataset. The goal is to use this data to create a data set that is as close to anomaly free as possible for a subsequent semi-supervised learning approach. The semi-supervised learning approach will be an autoencoder that would, after been trained on anomaly free data, be in a position to recreate normal observations, with abnormal observations identified by the autoencoder as anomalous. Figure 2.23 shows this data preparation flow graphically.

The first step is to decide how many observations are anomalous and need to be removed from the dataset. A balance between removing too many observations with the risk of also removing normal observations and removing too few with the risk of leaving anomalies is needed. The approach taken was to sort the observations based on the results for each of the 10 techniques. These outlier values are then plotted as shown in Figure 2.24. KNN, KNN node, KNN node/time, OCSVM node, OCSVM node/time, LOF and LOF node, all had a clear "knee" point in the region of 150 observations. OCSVM and MGD were not as clear with MGD showing a knee point at 2500. Based on this analysis it was decided to treat 150 points as anomalous.

The next step was to decide which 150 observations were to be removed. The first step in achieving this was to create ten separate lists of the observations, one for each of the approaches used. Each list was ordered from most anomalous to least, according the 10

Table 2.21: Common observations between the top 100 results per approach.

	KNN	KNN node	KNN node/time	OCSVM	OCSVM node	OCSVM node/time	LOF	LOF node	MGD node	MGD
KNN	100	94	92	96	91	90	61	57	38	83
KNN node	94	100	93	94	92	89	61	58	39	81
KNN node/time	92	93	100	93	98	95	60	55	38	78
OCSVM	96	94	93	100	92	91	58	54	37	80
OCSVM node	91	92	98	92	100	95	59	55	39	77
OCSVM node/time	90	89	95	91	95	100	58	53	34	75
LOF	61	61	60	58	59	58	100	74	39	65
LOF node	57	58	55	54	55	53	74	100	47	57
MGD node	38	39	38	37	39	34	39	47	100	42
MGD	83	81	78	80	77	75	65	57	42	100

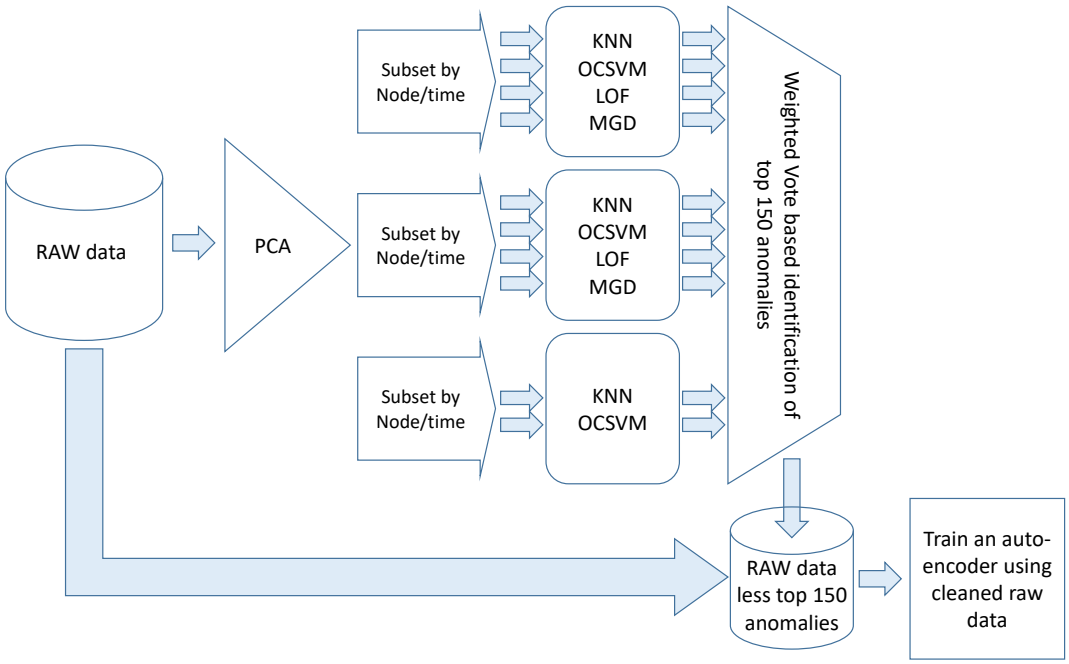
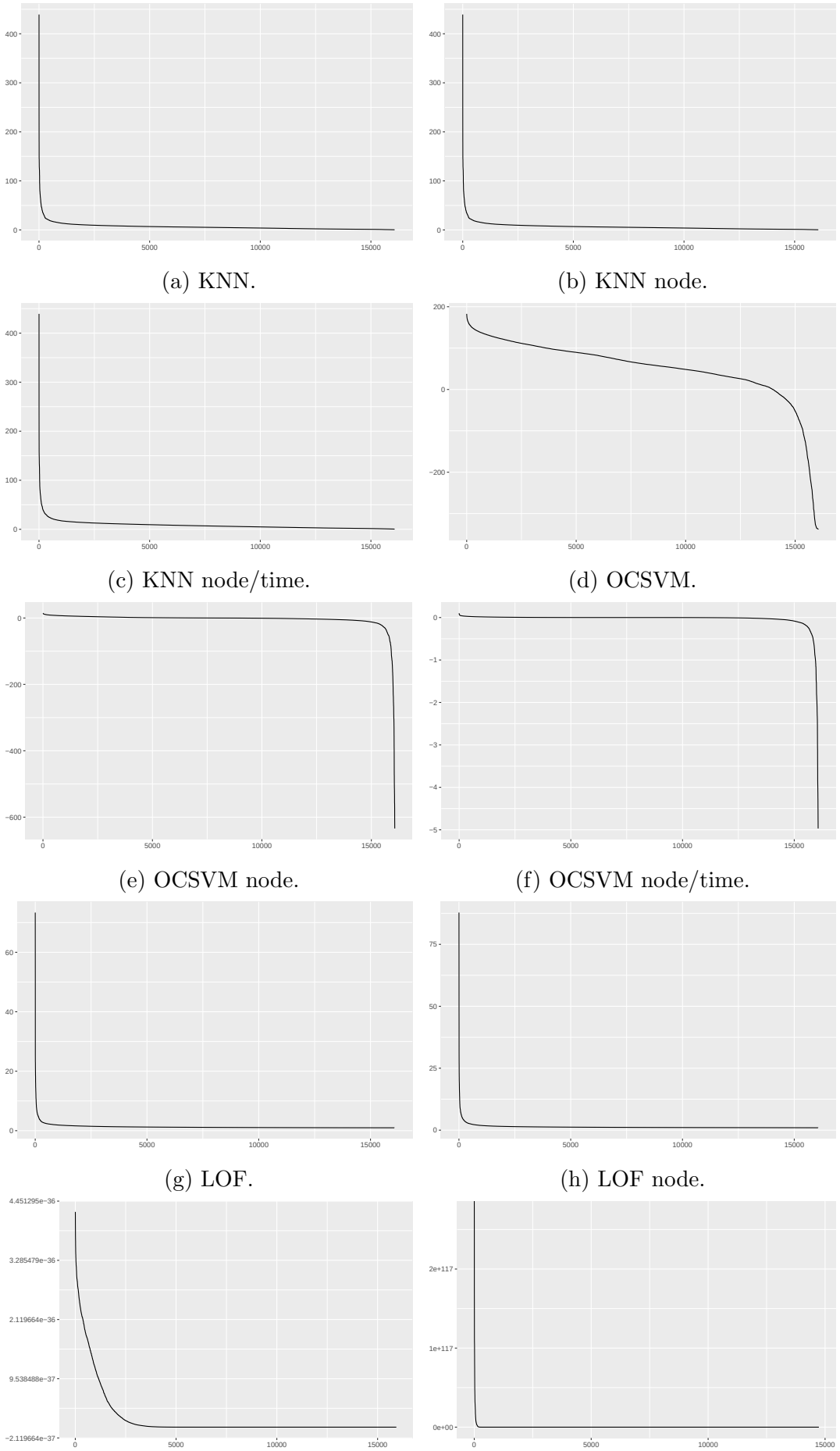


Figure 2.23: Data preparation process removing anomalies for autoencoder training.

different approaches used. Next a ranking number is added to each of the ten list, with 1 being assigned to the most anomalous and 16070 to the least anomalous. The 10 rankings for each observation were then added together to get a final ranking. Thus observations that received low ranking from all ten techniques had low overall final rankings. The 150 observations with the lowest final rankings were removed from the original raw data set. Table 2.22 shows the percentage of each techniques' top 150 that made the final anomaly list. The technique with the highest number of observations that made the final anomaly list was MGD with 98.7% and the least was MGD node with 50.7%. The three KNN approaches as a group identified the most anomalies. This resultant dataset formed the basis for training the final production anomaly detection model described in the next section.

Once the anomalous observations had been removed from the original raw dataset, the engineered features required for contextualizing the anomalies were added. The final data set thus consisted of 28011 observations with 1001 variables. These variables consisted of the SGW that generated the observation, the region that the SGW is located, the date/time stamp, the hour of the day, the time of day, the minute of the day day of the week, as well as 996 SGW counters.



(a) KNN.

(b) KNN node.

(c) KNN node/time.

(d) OCSVM.

(e) OCSVM node.

(f) OCSVM node/time.

(g) LOF.

(h) LOF node.

(i) MGD.

(j) MGD node.

Figure 2.24: Ordered Outlier factor values per Approach.

Table 2.22: The number of each techniques' top 150 that made the final anomaly list.

Approach	**Percentage Included in Final 150**
KNN.Weight50	84.7%
KNN.Node.Weight50	86.0%
KNN.Node.Time.Weight8	80.0%
OCSVM.ν.13.γ.0.002	81.8%
OCSVM.Node.ν.0.5.γ.1e.04	80.0%
OCSVM.Node.Time.ν.0.5.γ.1e.04	80.0%
LOF.K100	68.7%
LOF.Node.K100	64.7%
MGD.Node	50.7%
MGD	98.7%

Chapter 3

Autoencoder for Anomaly detection

3.1 Neural Network Based Approach

The next step in creating a anomaly detection system is to use the anomaly free data set, the creation of which is described in the previous chapter, to create an semi-supervised autoencoder. Before this is described, this chapter introduces neural networks in general and then focuses in on the autoencoder and the use of autoencoders for anomaly detection. Finally the details of the autoencoder implementation used in this project and the results obtained are detailed.

As described in [29] the history of the development of Artificial Neural Networks (ANN) span the better part of a century, with the foundations being laid in the 1940's in [30]. Later work in the 1960's included the perceptron convergence theorem detailed in [31] as well as [32] showing the shortcomings of a perceptron. This work resulted in interest being curbed until the 1980's where [33] introduced the "energy" approach and [34] first introduced the back propagation algorithm for multilayer networks.

Artificial neural networks are described as "weighted directed graphs in which artificial neurons are nodes and directed edges (with weights) are connections between neuron outputs and neuron inputs." in [29]. An example of a simple artificial neural network consisting of two hidden layers is given in Figure 3.1.

The leftmost side shows the input to the network with X_1 to X_n each representing an input variable. The input variables are all connected to the first hidden layer through weights which are applied to them. The nodes in the first hidden layer each take all the weighted inputs and sums them. These result are then used as inputs to the nodes' activation functions. The outputs of the activation functions will depend on the activations function used, but generally low input values are suppressed and high values are passed through. Various linear and non linear activation functions exist. Four sample activation functions are shown in Figure 3.2. These are the identity function Equation 3.1, the binary step Equation 3.2, the TanH function Equation 3.3 and the rectified linear unit Equation 3.4

$$f(x) = x \tag{3.1}$$

$$f(x) = \begin{cases} 0 \text{ for } x < 0 \\ 1 \text{ for } x \geq 0 \end{cases} \tag{3.2}$$

$$f(x) = \frac{(e^x - e^{-x})}{(e^x + e^{-x})} \tag{3.3}$$

$$f(x) = \begin{cases} 0 \text{ for } x < 0 \\ x \text{ for } x \geq 0 \end{cases} \tag{3.4}$$

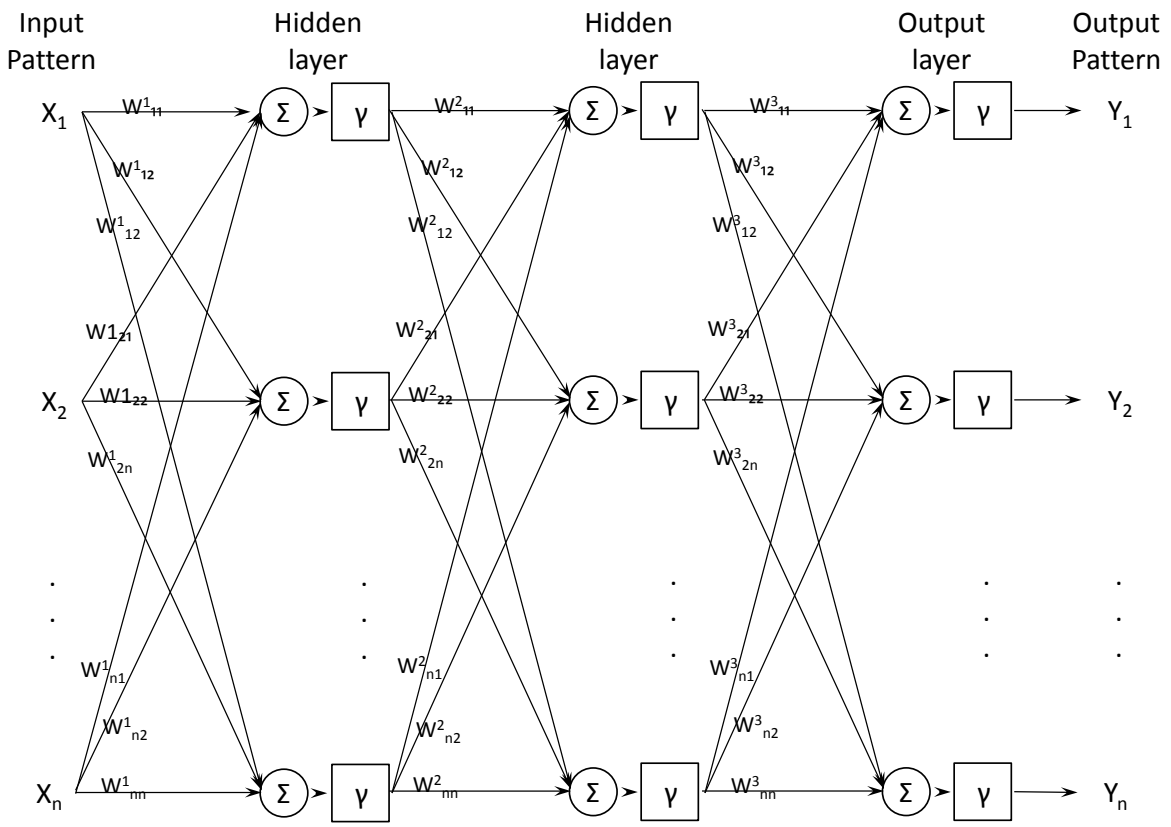
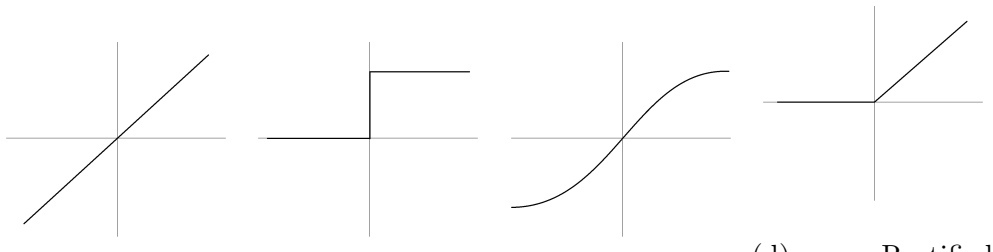


Figure 3.1: Neural network architecture with 2 hidden layers.

The outputs of the neurons in the first hidden layer are then passed on to the the next layer in the same way as the input variables were passed on to the first hidden layer. The number of hidden layers can vary. The last hidden layer then connects to the output layer, shown on the right of Figure 3.1 which produces the output. The initial weights in the network are chosen randomly and need to be changed to produce the required output. The iterative process of tuning the weights, introduced by [34] is called back-propagation. Back-propagation is described in [29] as an algorithm consisting of seven steps. The first step is to set the weights at the start to have small random values. The second step is to choose the first input pattern $X^{(\mu)}$. This initial pattern is then propagated through the network. The fourth step is to compute δ_i^L Equation 3.5 in the output layer $o_i = y_i^L$. h_i^u is the sum of the inputs to the i^{th} unit in the L^{th} layer, g is the activation function, with g' being the derivative. The fifth step is to calculate the deltas for the previous layers



(a) Identity. (b) Binary Step. (c) TanH. (d) Rectified Linear Unit.

Figure 3.2: Sample activation functions.

by propagating the errors back as in Equation 3.5 for $l = (L-1), ..., 1$. The weights are then updated using $\Delta W_{ji}^{i} = \mu \delta_{i}^{l} \epsilon_{j}^{l-1}$. This would complete the first iteration. The second iteration would start again at step two using the newly adjusted weights, and the next input pattern. This continues until either the output layer is within a required threshold or a selected maximum number of iterations is reached.

$$\delta_{i}^{L} = g'(h_{i}^{L})[d_{i}^{u} - y_{i}^{L}] \tag{3.5}$$

$$\delta_{i}^{l} = g'(h_{i}^{l}) \sum_{j} W_{ij}^{l+1} \delta_{j}^{l+1} \tag{3.6}$$

A specific structure of ANN called an autoencoder aims to represent the input at the output with the least possible distortion. Autoencoders were first introduced in [35]. The uses for autoencoders include dimensionality reduction. In this case, the number of neurons in the hidden layer is less than the number of neurons in the input and output layers. This forces the hidden layer to represent the input in a fewer number of dimensions. Another use for the antoencoder is anomaly detection. The concept of using the autoencoder for anomaly detection was introduced in [36]. The concept used is to train the autoencoder using only normal observations. In effect the autoencoder learns how to generalize the commonalities between the normal observations. When presented with a new normal observation, the autoencoder is expected to reconstruct this observation accurately at the output. If the autoencoder is presented with an observation that it is not able to reconstruct on the output then the point is considered to be an anomaly. The reason for this is that the anomalous data point does not conform to the commonalities that the autoencoder has learned to generalize. The difference between the input and the output is called the reconstruction error. The larger the reconstruction error the more likely that the point is an outlier. A benefit to this approach is that the model does not need anomalous data points in the training set. This benefit really comes into its own in instances that obtaining anomalous data points are expensive or difficult to obtain and in instances where not all types of anomalies are known upfront.

The autoencoder used in this study consisted of one hidden layer and an output layer. Both the hidden layer and the output layer had the same number of neurons which were equal to the number of input variables. The input variables included the 996 SGW counters and the input factors including day of the week, time of day, SGW node and region were expanded out to add another 162 inputs, totaling 1158. The activation function used was the TanH function shown in Figure 3.2 sub-figure c.

The dataset was divided into a test set, validation set and a training set each consisting of 10%, 10% and 80% of the original data set. After one epoch both the validation and training Mean Square Error of the autoencoder had dropped to 0.001 and after 4.8 epochs, both had reduced slightly more to 0.0008. 665 of the variables had no importance. This was expected due to the large number of variable consisting only of zeros. The remaining 493 variables covered a narrow range of percentage variable importances, with the most important variable being "Time.4:30" with a percentage of 0.2393% and the least being counter G19M4C8 having an importance of 0.1749%. These percentages are very low, showing that all variables contribute to the model. On average the group of variables consisting of the lowest average percentage importance were the counters, with the variable containing the time of day having the highest. The average percentages for the various groups are shown in Table 3.1.

Table 3.1: Average percentage variable importance per variable group

Variable Group	Average Percentage Variable Importance
SGW ID	0.225
Time of the day	0.223
Hour of the day	0.218
Day of the Week	0.212
Region	0.205
Minute of the day	0.198
Counter	0.194

3.2 Analysis of the New Data Set

The effectiveness of the autoencoder created through the process described above is detailed here. The process of reviewing its effectiveness entailed running a new raw data set obtained from the mobile network through the autoencoder. The resultant outliers were identified by having high reconstruction errors. These outliers were then compared to ground truth data describing real occurrences that took place in the mobile network. To contextualize the analysis, changes in the network that took place in the 5 month period between the first data extract and this one are pointed out as follows. Firstly the total volume of traffic supported by the SGWs increased by 27%. Secondly, old SGWs were replaced by new SGWs added to the network. S1u traffic was shifted from GGCF01 to GGCF02, from GGPR01 to NFV-GGPR02, from GGJF01 to NFV-GGMD01, from GGDM01 to GGDM02 and from GGDN01 to GGDN03.

3.3 Data Description

The new data set was obtained using the same SQL query used to select the original data set described in chapter 1. This returned all the columns from all the SGW tables containing the 15 minute data and joined them on SGW ID and Time of day. The data covered two week period starting at 09h30 on the 22nd of June 2018 and ending at 08h30 on the 6th of July 2018.

3.4 Results

This new dataset was fed through the autoencoder and the reconstruction MSE per observation were analyzed. The average MSE for all the observations was 0.032, with the highest value being 96.57978 and the lowest being 0.000154722. The analysis of these results are grouped into 3 groups for discussion purposes. The first group includes the 196 observation that had a reconstruction MSE higher than 1. The second group are all the observations with an MSE below 0.002. The last group consistis of the observations having reconstruction MSEs between 0.002 and 1.

3.4.1 Reconstruction MSE Greater Than One

The anomalies with a reconstruction MSE greater than 1 are shown in sub-figure (a) of Figure 3.3. Closer inspection of these points reveal five different groups of anomalies. The first are the group of anomalies that occurred between 09h30 on the 22nd and 17h00

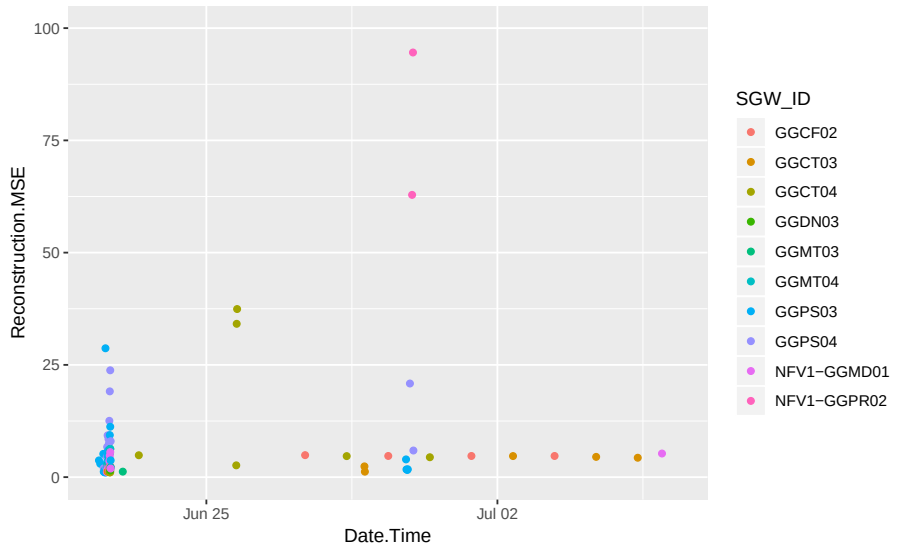
on the same day. Initially, SGW GGPS03 shows an increase in MSE values from 09h30 this was followed by all SGWs showing an increase in MSE until 17h00. These anomalies account for 151 of the 196 anomalies with a reconstruction MSE higher than 1. These are shown in sub-figure (b) of Figure 3.3. When looking at the ground truth of what was happening on the network to cause this it is apparent that there was a once off data promotion that started at 09h00 on that day and finished at midnight. The promotion increase traffic dramatically. Due to the increase in traffic data services using QCI 1 and QCI 5 started to experience congestion between 09h40 to 12h10. From 12h50 to 17h00 the mobility management of LTE based services experienced a degradation. This LTE mobility management degradation aligns well with the jump in reconstruction MSEs for all the SGWs at the same period. When comparing the counters from this anomalous interval to and equivalent anomaly free interval, an increase of between 21 and 2317 times is evident on the following counters. A concurrent increase in these counters could be used to identify a degradation in the LTE network performance due to high user plane load.

- G19M1C22, Total PDNs Released with local reason
- G19M4C22, Total Idle-mode TAU Inter-SGW handover failed
- G19M4C33, Total Inter-MME Intra-SGW handover failed
- G19M1C27, Total PDNs Released with reason S5 Path Failure on SGW (part of SAEGW)

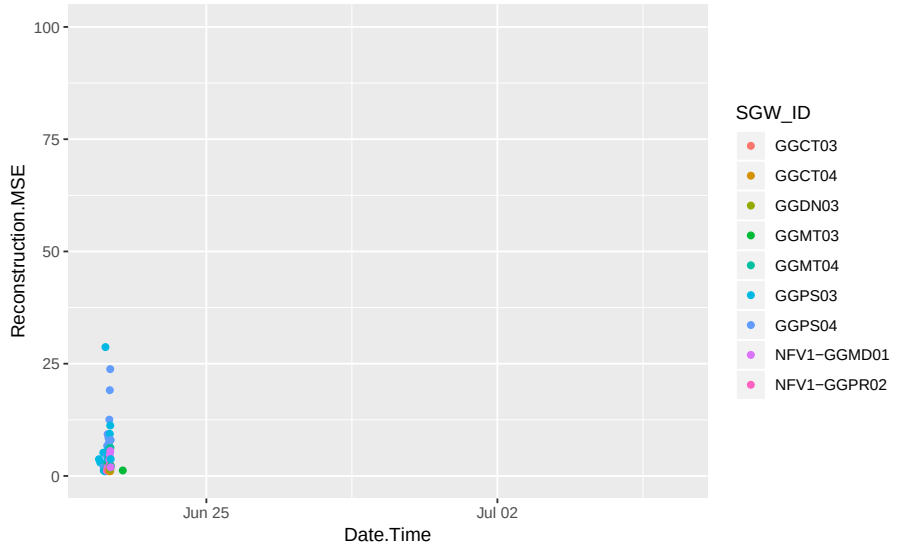
The second of the five clusters is shown in sub-figure (c) of Figure 3.3. There is a single anomaly that appears randomly on one of the CTN SGWs on 10 of the 14 days. Each day the reconstruction MSE has approximately the same value, between 4 and 5. To investigate the reason for these anomalies, the 09h00 observations with high reconstruction MSEs and the 09h15 observations with low reconstruction MSEs in the CTN region were compared. This investigation shows that only 4 counters differed dramatically between the 09h00 and the 09h15 intervals. These are shown in the list below. The 09h00 anomalous observations were on average 40 to 155 times higher than the non anomalous 09h15 observations for these four counters. Further investigation revealed that there was one PDN being set up of the type ipv4v6 during these anomalous observations. None of the other observations in the entire data set have this type of PDN. This indicated that there was only one subscriber or Machine to Machine device that would set up a PDN at 09h00 every day in the CTN region and then transfer data for a short period and then delete the PDN shortly afterwards. The MME pooling resulted in the random selection of a CTN SGW. A concurrent increase in these counters could be used to identify an increase in ipv4v6 traffic in future.

- G19M18C67, ipv4v6 pdn ipv4 to user packet
- G19M18C68, ipv4v6 pdn ipv4 to user byte
- G19M18C69, ipv4v6 pdn ipv4 from user packet
- G19M18C70, ipv4v6 pdn ipv4 from user byte

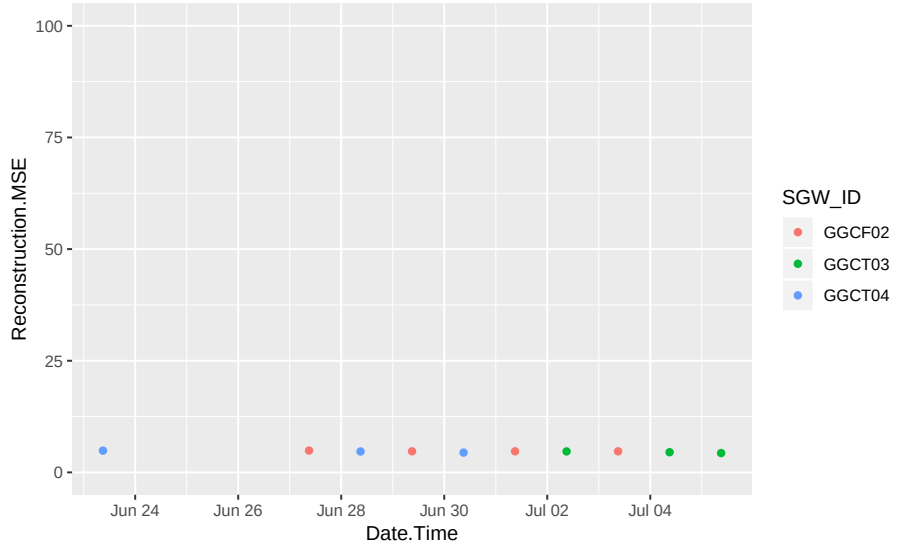
The third of the five clusters is shown in sub-figure (a) of Figure 3.4. This shows three successive intervals with high reconstruction MSEs on GGCT04 on the 25th from 17h15 to 17h45. An investigation into the normal intervals on GGCT04 just prior to this reveal



(a) Consolidated View.

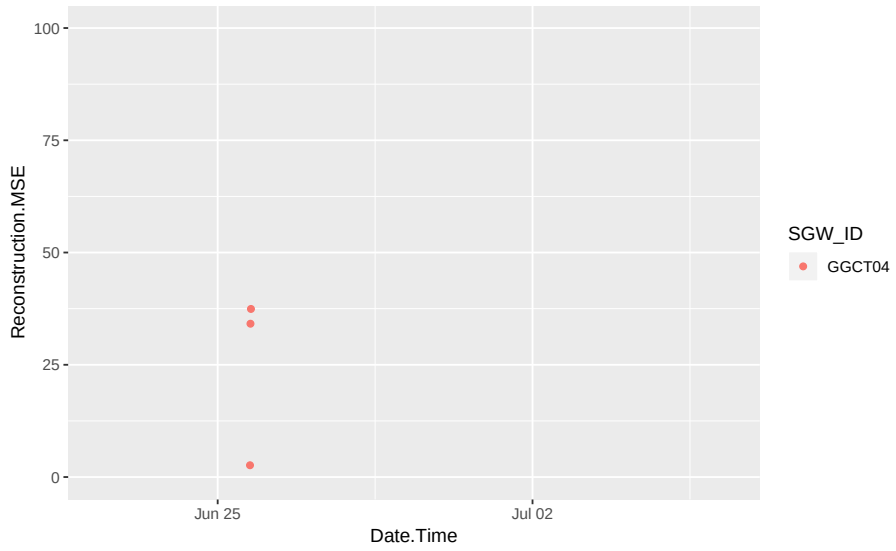


(b) First Promotion Cluster.

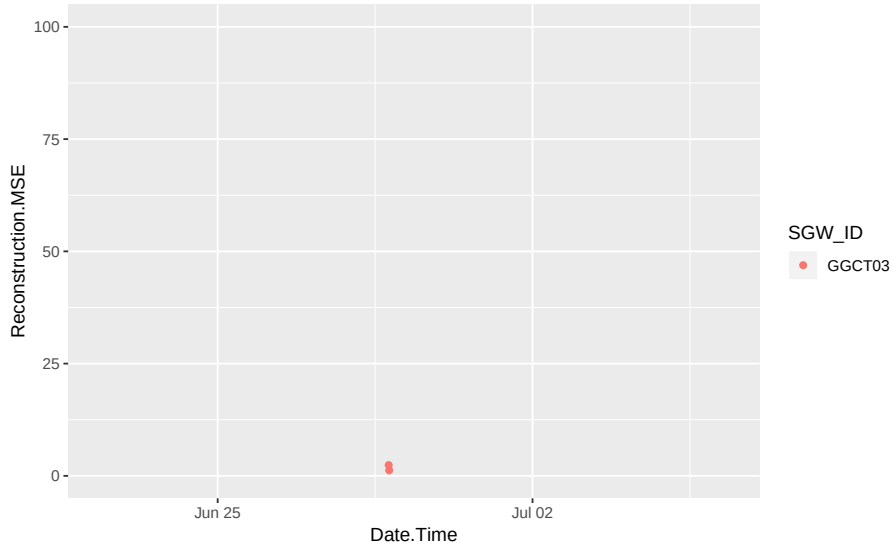


(c) CTN 09h00 Cluster.

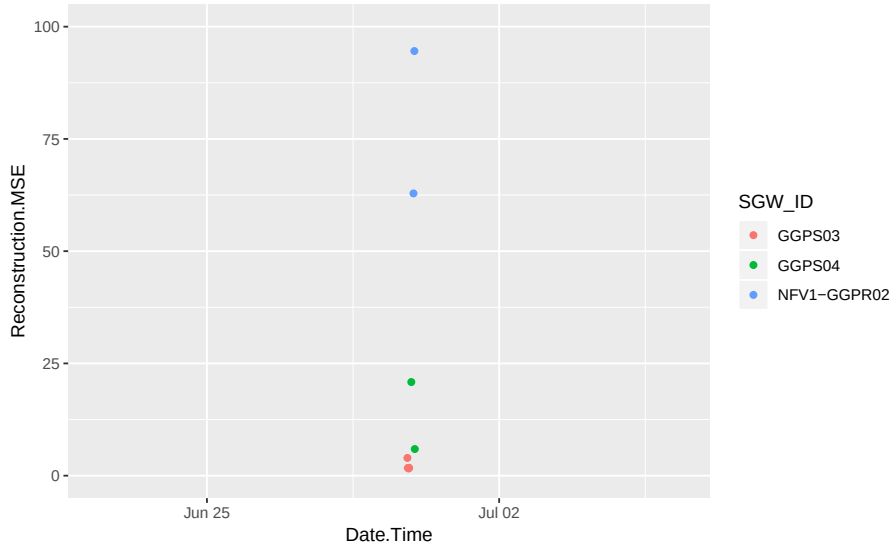
Figure 3.3: Anomalies with a reconstruction MSE greater than 1.



(a) GGCT04 Cluster.



(b) GGCT03 Cluster.



(c) Second Promotion Cluster.

Figure 3.4: Anomalies with a reconstruction MSE greater than 1.

that 4 counters were from 214 and time to 464 times higher during the anomalous interval. These counters are listed below. This points to jump in traffic generated by an inbound roamer using a QCI5 service. The only other period where there is a spike in these counters is at 19h30 and 20h00 on the 28th of June on GGCT03. This comes up as the forth cluster of anomalies and is show in sub-figure (b) of Figure 3.4. A concurrent increase in these counters could be used to identify the use of QCI5 services by an inbound roamer in future.

- G19M11C3, s8 uplnk qci5totbyte
- G19M11C4, s8 uplnk qci5totpkt
- G19M11C43, s8 downlnk qci5totbyte
- G19M11C44, s8 downlnk qci5totpkt

The final of the five clusters is shown in sub-figure (c) of Figure 3.4. This cluster correlates with a second data promotion that took place from 09h00 to midnight on the 29th of June. The increase in traffic resulting from this promotion was less than that generated by the first promotion. To determine the underlying difference in that caused the observations from NFV1-GGPR02 at 22h45 and 23h15 on the 29th to be anomalous, they were compared to the intervals before, between and after, which had low reconstruction MSE of 0.006, 0.001 and 0.02 respectively. One counter, G19M1C27, which counts the total PDNs released with reason "S5 Path Failure on SGW" showed a distinct difference. Through the two weeks observed, this counter had a mean value of 0.36 and had a maximum value of 8 for all intervals, excluding the two anomalous intervals, which has counts of 1090 and 889.

3.4.2 SGWs With Reconstruction MSE Less Than 0.002

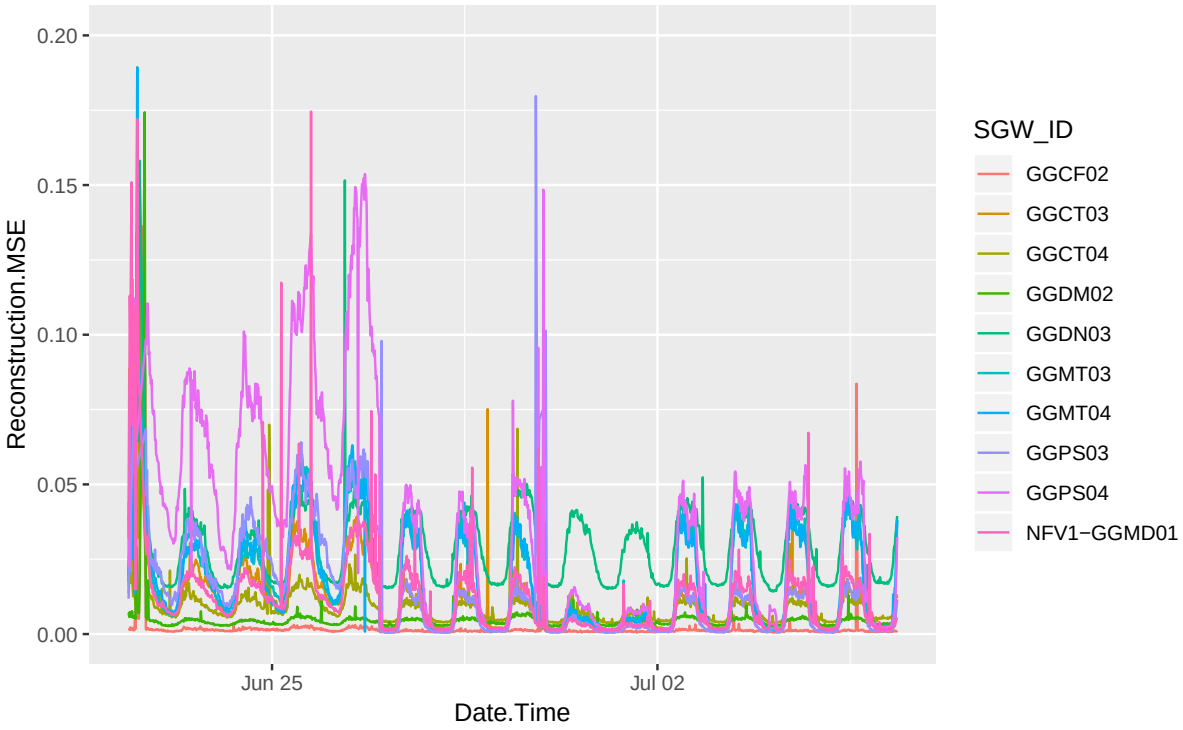
The next major group of observations discussed are on at the other end of the spectrum. These are the observations associated with SGWs where all observations have a reconstruction MSE below 0.002. These are shown in Figure 3.5. These SGW carried no traffic during the collection of the test dataset. It is noted that the following SGWs carried no traffic during the training interval GGCT01, GGCT02, GGMT01 and GGPS02. It was expected that the model would reproduce these observations accurately. The following SGWs did carry traffic during the training period, GGDM01, GGCF01, GGPR01, GGDN01 and GGJF01. It is interesting to note that a total cessation of traffic on these nodes was not considered abnormal by the model.

Figure 3.5: SGWs with reconstruction MSEs below 0.002.

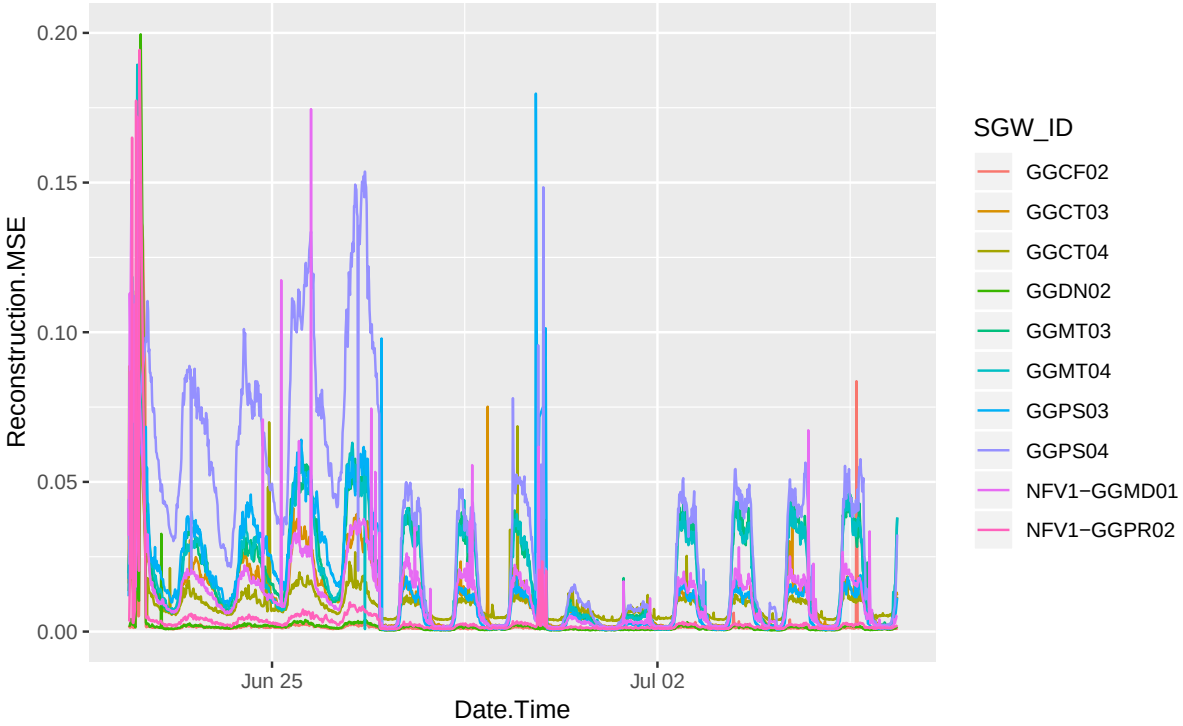
3.4.3 Reconstruction MSEs Less Than One

The last group of observations discussed include those remaining with reconstruction MSEs between zero and 1. An overview of these are shown in sub-figure (a) of Figure 3.6. During the congestion experienced on the 22nd, the the support of VoLTE including the QCI1 and QCI5 services were deactivated. This resulted in the QCI1 and QCI5 related counters to drop to zero. These counters are listed in Table 3.2. These services were reactivated on the 26th of June at 23h00. Sub-figure (b) of Figure 3.6 shows the SGWs that had a reduction in reconstruction MSE at this time. This is compared to the SGWs in sub-figure (b) of Figure 3.7 where the resumption of the QC1 based services didn't impact the reconstruction MSE. Sub-figure (a) of Figure 3.7 shows the reconstruction MSEs for the 5 new SGW introduced after the initial dataset collection. Two of the nodes, GGDM03 and GGDN02 showed no reaction to the resumption of the QCI1 and QCI5 services, where as NFV-GGPR02, NFV-GGMD01 and GGCF02 had a reduction in the reconstruction MSE after the services were resumed.

The question as to why the cessation of VoLTE services has such a small impact on the reconstruction error needs to be asked. The first thought was that there might have been examples of SGWs in the training set which supported LTE Data services but not VoLTE. Alternatively there could have been intervals where there were LTE Data services, but no VoLTE services. Both of these examples would have exposed the autoencoder during training to scenarios similar to those seen in the first 5 days of the test data set. This would have explained why the cessation of VoLTE services had such a little impact of the reconstruction MSE. But investigation into the training data set revealed neither of these two scenarios. An alternative reason could be that high levels of correlation between VoLTE related counters led the autoencoder to learn to treat all these counters as a feature set in the data.

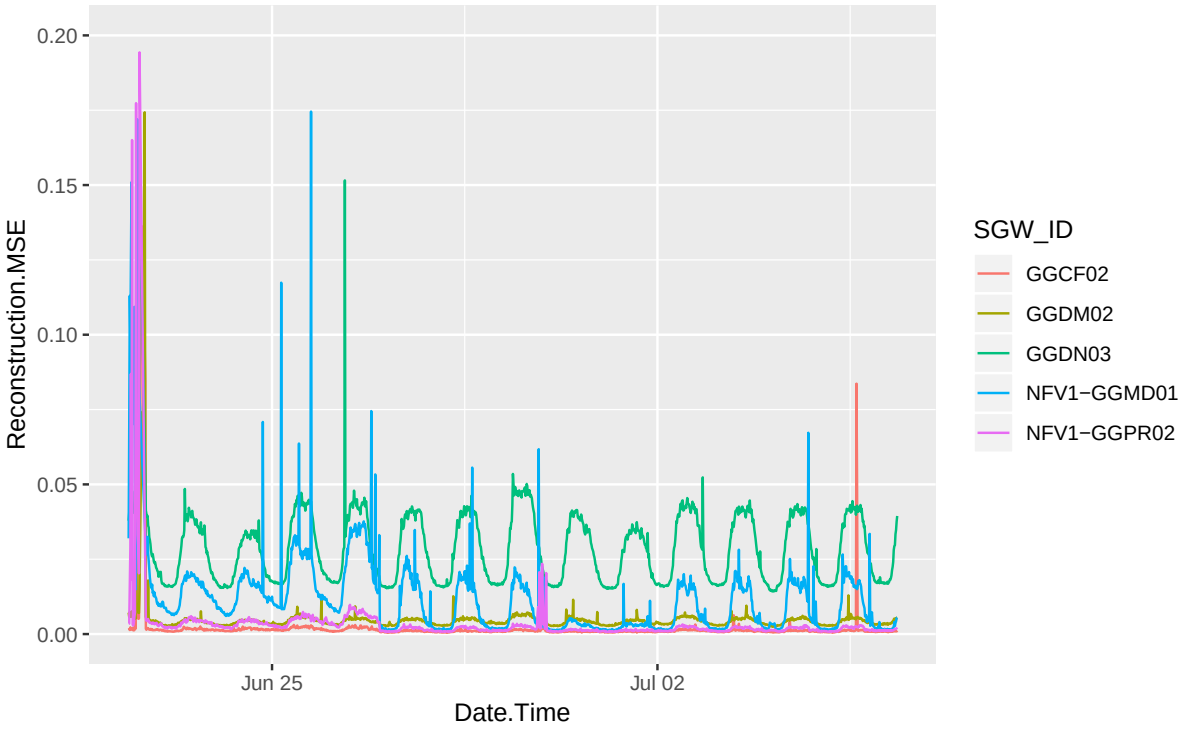


(a) Consolidated View.

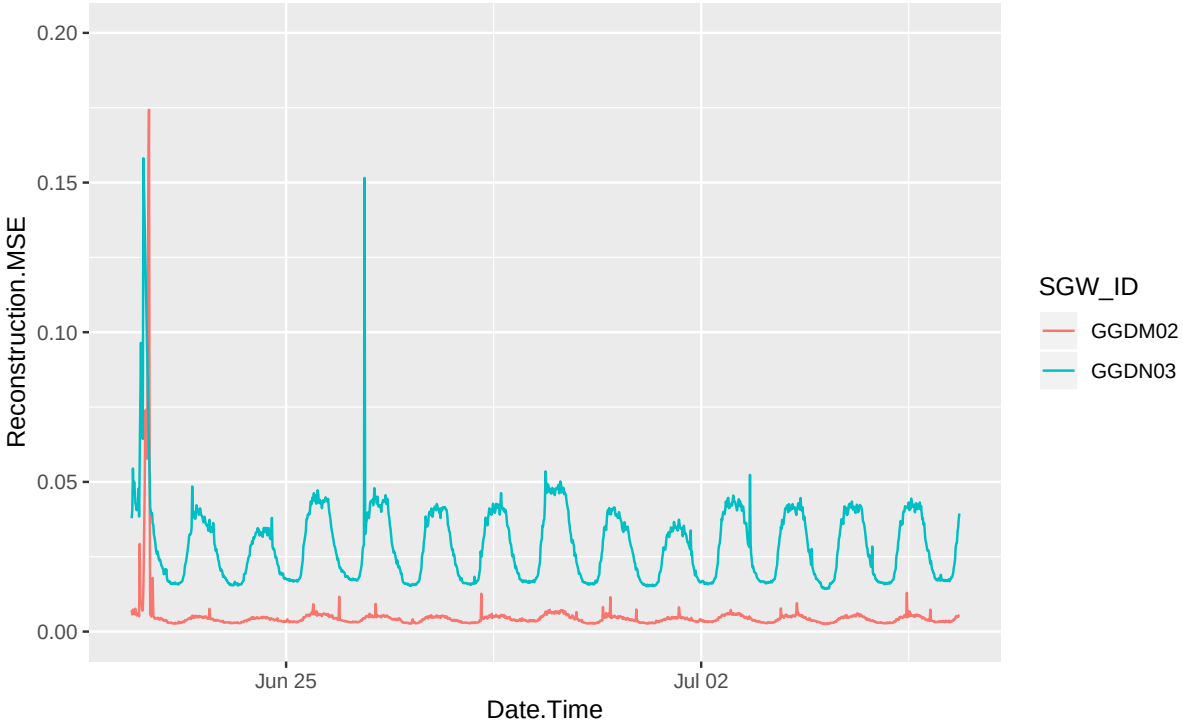


(b) VoLTE Sensitive.

Figure 3.6: Observations with an Reconstruction MSE below 0.2.



(a) New SGW Nodes.



(b) VoLTE insensitive.

Figure 3.7: Observations with an Reconstruction MSE below 0.2.

Table 3.2: QCI 1 and 5 related counters.

QCI 1 and 5 related counters
G19M10C5, s5 downlnk qci1totbyte
G19M10C6, s5 downlnk qci1totpkt
G19M12C24, s5s8 uplnk qci1totbyte
G19M12C25, s5s8 uplnk qci1totpkt
G19M13C10, s5s8 downlnk qci1totpkt
G19M13C9, s5s8 downlnk qci1totbyte
G19M1C50, Total EPS Bearers Released QCI 1
G19M2C12, QCI 1 Total Bytes
G19M2C13, QCI 1 Total Packets
G19M2C3, Total EPS Dedicated Bearers Released Reason PGW Initiated
G19M4C54, s1u uplnk qci1totbyte
G19M4C55, s1u uplnk qci1totpkt
G19M5C39, s1u downlnk qci1totbyte
G19M5C40, s1u downlnk qci1totpkt
G19M9C23, s5 uplnk qci1totbyte
G19M9C24, s5 uplnk qci1totpkt

3.5 Usability in Industry

The aim of this project is to create a system that can augment the current model based performance management systems deployed in the network. It is in this context that the value of the results are discussed here.
The results discussed in section 3.4 show that the autoencoder is able to highlight both network wide anomalies, affecting many counters, such as the event experienced during the first promotion, as well as more subtle anomalies such as those caused by single subscribers. The major benefit is that no prior network knowledge was required to create the anomaly detection system. Added to this the system was able to take over a thousand counters in and create a model that monitored them all. It is this blind spot that the system was hoping to cover. .y On the negative side, the system provides no information into why points are considered anomalous. The black-box nature of the autoencoder gives no insight into what caused an observation to be considered an anomaly. It must also be noted that the results only provide insight into whether the system is performing in a similar way to the way it was performing when the training data set was collected. Obviously much effort went into removing anomalies, but the remaining training dataset could have been from a system that was running sub-optimally. An increase in reconstruction MSEs therefore only indicates that there was a change in the system. This could be a change for the better or a change for the worse. This change can only be used as a trigger to the system user to investigate what the change was and whether or not this results in an improvement or a degradation in the service offered by the system. It is a pity that the system is therefore not capable of assisting in the investigation into why a point was considered anomalous.
There was a five month interval between the initial data collection used to train the model and the the final data collection used to test the model. The average reconstruction error was still low at 0.032. This longevity of the model created is beneficial, as the efforts associated with regular model updates are reduced.

Chapter 4

Conclusion

An approach to detecting anomalies in high dimensional data of the SGWs in the mobile network environment was successfully implemented. This augments the current model based approach which detects degradations in predetermined KPIs. This anomaly detection approach fills the blind spot left by the current model based approach, covering all counters. The approaches ability to pick up anomalies affecting many subscribers and counters, as in the case of the first promotion, as well as being able to pick up anomalies introduced by single subscriber activities affecting few counters was demonstrated. The models ability to treat a group of counters as being related to a service supported by the mobile network as a feature in the data was evident in its handling of the VoLTE service cessation and resumption. The benefit of this being that an in-depth understanding of the service and the associated counters is not required, as it would have been in the traditional model based approach.
Four different approaches were used to identify anomalies for removal from the original training data set. Even though the four approaches differed fundamentally, there was a high level of agreement between them as to which observations were anomalies. It must be noted that even after removing all the anomalies, the training data could still represent a suboptimal operating state of the mobile network. Anomalies highlighted by the approach only indicate a change from this baseline. The changes could represent a degradation or and improvement in the performance of the network. Further investigation into the mobile network is required to determine which of the two it is and what the underlying details of the changes are. Unfortunately the black-box nature of the final auto encoder gives no assistance to the user in these investigations.

4.1 Future Work

This study takes the time of day for each observation into account, but each observation is considered independent of the rest. In reality, the system generating this data is a continuous system, with the values from one time interval following on from the previous. The anomaly detection system could benefit from taking this into account. In a similar vein, each observation represents the data for one SGW. As these SGW work together, with load shifting from one to the other, the anomaly detection system could benefit from expanding the observation variable set to be a network wide one. Taking this theme further, the SGW is just one element type in the entire network, expanding the observation definition to include variables from other element types would make it possible to give a network wide view. Future work could investigate the benefit that this brings as well as the impact of moving from catering for observations containing thousands of variables to millions of variables. As mentioned, the black-box nature of the autoencoder, results

in the user getting no guidance into what the underlying cause for the anomaly might be. Future work could look at augmenting this anomaly detection approach with another approach that provides insight into the counter changes that triggered an anomaly to be flagged.

Appendix A

SQL For Data Extraction

This is a partial view of the SQL code used to extract the data from the SGW tables. The first 12 and the last 3 counters of the 1011 counters are shown.

```
SELECT
    SGWG.STARTTIME
    ,SGWG.REGION
    ,SGWG.SGW_ID
    ,SUM(SGWA.G19M10C1) as  G19M10C1  --  s5_uplnk_drop_qci9totbyte
    ,SUM(SGWA.G19M10C10) as   G19M10C10 --  s5_downlnk_qci3totpkt
    ,SUM(SGWA.G19M10C11) as   G19M10C11 --  s5_downlnk_qci4totbyte
    ,SUM(SGWA.G19M10C12) as   G19M10C12 --  s5_downlnk_qci4totpkt
    ,SUM(SGWA.G19M10C13) as   G19M10C13 --  s5_downlnk_qci5totbyte
    ,SUM(SGWA.G19M10C14) as   G19M10C14 --  s5_downlnk_qci5totpkt
    ,SUM(SGWA.G19M10C15) as   G19M10C15 --  s5_downlnk_qci6totbyte
    ,SUM(SGWA.G19M10C16) as   G19M10C16 --  s5_downlnk_qci6totpkt
    ,SUM(SGWA.G19M10C17) as   G19M10C17 --  s5_downlnk_qci7totbyte
    ,SUM(SGWA.G19M10C18) as   G19M10C18 --  s5_downlnk_qci7totpkt
    ,SUM(SGWA.G19M10C19) as   G19M10C19 --  s5_downlnk_qci8totbyte
    ,SUM(SGWA.G19M10C2) as  G19M10C2  --  s5_uplnk_drop_qci9totpkt
                              .
                              .
                              .
    ,SUM(SGW9.G19M9C7) as   G19M9C7 --  s12_downlnk_drop_qci7totbyte
    ,SUM(SGW9.G19M9C8) as   G19M9C8 --  s12_downlnk_drop_qci7totpkt
    ,SUM(SGW9.G19M9C9) as   G19M9C9 --  s12_downlnk_drop_qci8totbyte

FROM
     STARENT_PSCORE_SA.GGSN_SGWG_STATS SGWG
    ,STARENT_PSCORE_SA.GGSN_SGW1_STATS SGW1
    ,STARENT_PSCORE_SA.GGSN_SGW2_STATS SGW2
    ,STARENT_PSCORE_SA.GGSN_SGW3_STATS SGW3
    ,STARENT_PSCORE_SA.GGSN_SGW4_STATS SGW4
    ,STARENT_PSCORE_SA.GGSN_SGW5_STATS SGW5
    ,STARENT_PSCORE_SA.GGSN_SGW6_STATS SGW6
    ,STARENT_PSCORE_SA.GGSN_SGW7_STATS SGW7
    ,STARENT_PSCORE_SA.GGSN_SGW8_STATS SGW8
    ,STARENT_PSCORE_SA.GGSN_SGW9_STATS SGW9
    ,STARENT_PSCORE_SA.GGSN_SGWA_STATS SGWA
```

```
    ,STARENT_PSCORE_SA.GGSN_SGWB_STATS SGWB
    ,STARENT_PSCORE_SA.GGSN_SGWC_STATS SGWC
    ,STARENT_PSCORE_SA.GGSN_SGWD_STATS SGWD
    ,STARENT_PSCORE_SA.GGSN_SGWE_STATS SGWE
    ,STARENT_PSCORE_SA.GGSN_SGWF_STATS SGWF


WHERE
    SGWG.STARTTIME > SYSDATE -14
    AND SGWG.STARTTIME < SYSDATE
    AND SGW1.STARTTIME = SGWG.STARTTIME
    AND SGW2.STARTTIME = SGWG.STARTTIME
    AND SGW3.STARTTIME = SGWG.STARTTIME
    AND SGW4.STARTTIME = SGWG.STARTTIME
    AND SGW5.STARTTIME = SGWG.STARTTIME
    AND SGW6.STARTTIME = SGWG.STARTTIME
    AND SGW7.STARTTIME = SGWG.STARTTIME
    AND SGW8.STARTTIME = SGWG.STARTTIME
    AND SGW9.STARTTIME = SGWG.STARTTIME
    AND SGWA.STARTTIME = SGWG.STARTTIME
    AND SGWB.STARTTIME = SGWG.STARTTIME
    AND SGWC.STARTTIME = SGWG.STARTTIME
    AND SGWD.STARTTIME = SGWG.STARTTIME
    AND SGWE.STARTTIME = SGWG.STARTTIME
    AND SGWF.STARTTIME = SGWG.STARTTIME

    AND SGW1.SGW_ID = SGWG.SGW_ID
    AND SGW2.SGW_ID = SGWG.SGW_ID
    AND SGW3.SGW_ID = SGWG.SGW_ID
    AND SGW4.SGW_ID = SGWG.SGW_ID
    AND SGW5.SGW_ID = SGWG.SGW_ID
    AND SGW6.SGW_ID = SGWG.SGW_ID
    AND SGW7.SGW_ID = SGWG.SGW_ID
    AND SGW8.SGW_ID = SGWG.SGW_ID
    AND SGW9.SGW_ID = SGWG.SGW_ID
    AND SGWA.SGW_ID = SGWG.SGW_ID
    AND SGWB.SGW_ID = SGWG.SGW_ID
    AND SGWC.SGW_ID = SGWG.SGW_ID
    AND SGWD.SGW_ID = SGWG.SGW_ID
    AND SGWE.SGW_ID = SGWG.SGW_ID
    AND SGWF.SGW_ID = SGWG.SGW_ID

    AND SGW1.VPN_NAME = SGWG.VPN_NAME
    AND SGW2.VPN_NAME = SGWG.VPN_NAME
    AND SGW3.VPN_NAME = SGWG.VPN_NAME
    AND SGW4.VPN_NAME = SGWG.VPN_NAME
    AND SGW5.VPN_NAME = SGWG.VPN_NAME
    AND SGW6.VPN_NAME = SGWG.VPN_NAME
    AND SGW7.VPN_NAME = SGWG.VPN_NAME
    AND SGW8.VPN_NAME = SGWG.VPN_NAME
```

```
    AND SGW9.VPN_NAME = SGWG.VPN_NAME
    AND SGWA.VPN_NAME = SGWG.VPN_NAME
    AND SGWB.VPN_NAME = SGWG.VPN_NAME
    AND SGWC.VPN_NAME = SGWG.VPN_NAME
    AND SGWD.VPN_NAME = SGWG.VPN_NAME
    AND SGWE.VPN_NAME = SGWG.VPN_NAME
    AND SGWF.VPN_NAME = SGWG.VPN_NAME

    AND SGW1.SERV_ID = SGWG.SERV_ID
    AND SGW2.SERV_ID = SGWG.SERV_ID
    AND SGW3.SERV_ID = SGWG.SERV_ID
    AND SGW4.SERV_ID = SGWG.SERV_ID
    AND SGW5.SERV_ID = SGWG.SERV_ID
    AND SGW6.SERV_ID = SGWG.SERV_ID
    AND SGW7.SERV_ID = SGWG.SERV_ID
    AND SGW8.SERV_ID = SGWG.SERV_ID
    AND SGW9.SERV_ID = SGWG.SERV_ID
    AND SGWA.SERV_ID = SGWG.SERV_ID
    AND SGWB.SERV_ID = SGWG.SERV_ID
    AND SGWC.SERV_ID = SGWG.SERV_ID
    AND SGWD.SERV_ID = SGWG.SERV_ID
    AND SGWE.SERV_ID = SGWG.SERV_ID
    AND SGWF.SERV_ID = SGWG.SERV_ID

GROUP BY
    SGWG.STARTTIME
    ,SGWG.REGION
    ,SGWG.SGW_ID
```

Appendix B

r Code

The following lists the r code [20] used in through this project

B.1 SGW data investigation

```
---
title: "SGW data investigation"
author: "Jason Salzwedel"
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
rm(list=ls())
```

Read in SGW data

The first step is to read the SGW data in and assign it to a variable.
There are two options to do this. The first is to open a connection up
from R with the Oracle database where the data is stored and then to
run an SQL query to collect the data. The second option is to query the
Oracle database using SQL developer and to save the results to a csv
file. This file can then be read into an R data frame.

Direct Oracle connection

```{r DBconnection echo = FALSE}

library(ROracle)
library(readr)
library(ggplot2)

#set up an connection to the oracle database
drv <- dbDriver("Oracle")
con <- dbConnect(drv, dbname="OPTPROD", user="salzweja", password="xxxxxxxxx")

queryLoc <- "../SQL/SGW.sql" #select the SQl to be run
```
```

```
query <- read_file(queryLoc)

Data<- dbGetQuery(con, query) # connect to the oracle DB and run the query
dbDisconnect(con)
```


```
# Reading data in from file

```{r DBconnection echo = FALSE}

Data <- read.csv("../SQL/SQLResults/SGW14days.csv", header = TRUE, sep = ",")
Raw.Data.tib <- Data
```


```
## Data exploration

```{r Flip function, echo=FALSE}

# This function flips matrix to bitmap representation
flip.over <- function(x){
  flipped <- x
  for (i in 1:ncol(x)){
    flipped[,i]<- x[,ncol(x)-i+1]
  }
  return(flipped)
}
```


```
# Tidy data

```{r Tidyverse, echo=FALSE}

library(tidyverse)

# 1) find the N/A per count per variable and delete variables above a
#    manually decided level.
na_count <-sapply(Raw.Data.tib, function(y) sum(length(which(is.na(y)))))

#na_count.sorted <- (sort(unname(na_count)))
# Show the how many Variables per N/A count. Use this table to decide
# how many variables to delete
table(na_count)
# Remove Variables the greater than 11 N/As
index <- unname(na_count)<11

Data.less.na.var <- Data[,index]
```
```

```
# 2) Remove observations with N/A
Data.less.na.observation <- Data.less.na.var[complete.cases(Data.less.na.var),]
# how many rows were removed?
row.has.na <- apply(Data.less.na.var, 1, function(x){any(is.na(x))})
sum(row.has.na)


# 3) Remove variables containing only zeros
# Create the index for the 'ZERO' variables
zero.index <- which(colSums(unname(Data.less.na.observation)[,c(-1,-2,-3)])==0)
Data.tib.less.zero <- Data.less.na.observation[,-((zero.index)+3)]
zero.index

# 4) Remove data for test nodes and non LTE Node

SGW.List <- c("GGCF01","GGCT03","GGCT04","GGDM01","GGDN01","GGDN02","GGJF01"
  ,"GGMT03","GGMT04","GGPR01","GGPS03","GGPS04")
Clean.Data <- Data.tib.less.zero %>% filter(SGW_ID %in% SGW.List)
#Investigate the impact of QCI differences on the covariance matrix
```


```{r Feature enigeering , echo=FALSE}

# add features to the data set to modify contextual anomaly detection to
# point anomaly detection
library(lubridate)
library(plotly)
Final.Data.SGW <- Clean.Data %>%
  mutate(Date.Time = as.POSIXct(as.character(levels(Clean.Data$STARTTIME)),
    format = "%Y-%m-%d %H:%M")[Clean.Data$STARTTIME]) %>%
  mutate(Day = wday(Date.Time))%>%
  mutate(Hour = hour(Date.Time))%>%
  mutate(Qtr = minute(Date.Time))%>%
  mutate(Qtr = replace(Qtr, Qtr==0,"00"))%>%
  mutate(Time = paste(Hour,Qtr, sep = ":"))%>%
  mutate(Day = as.factor(Day))%>%
  mutate(Hour = as.factor(Hour))%>%
  mutate(Qtr = as.factor(Qtr))%>%
  mutate(Time = as.factor(Time))%>%
  select(Time,Qtr,Hour,Day,Date.Time, everything())
```


```{r Save off Final Data , echo=FALSE}

write.csv(Final.Data.SGW,file = "FinalDataSGW.CSV")
```

```{r min max mean median var sd , echo=FALSE}

Data.sd <- Final.Data.SGW[,-c(1:8)] %>% summarise_all(sd)
Data.sd.gather <- gather(Data.sd)
ggplot(Data.sd.gather, aes(value)) + geom_histogram()
#ggsave("SGWSD.pdf")

Data.var <- Final.Data.SGW[,-c(1:8)] %>% summarise_all(funs(var))
Data.var.gather <- gather(Data.var)
ggplot(Data.var.gather, aes(value)) + geom_histogram()
#ggsave("SGWvar.pdf")

Data.max <- Final.Data.SGW[,-c(1:8)] %>% summarise_all(funs(max))
Data.max.gather <- gather(Data.max)
ggplot(Data.max.gather, aes(value)) + geom_histogram()
#ggsave("SGWmax.pdf")

Data.min <- Final.Data.SGW[,-c(1:8)] %>% summarise_all(funs(min))
Data.min.gather <- gather(Data.min)
ggplot(Data.min.gather, aes(value)) + geom_histogram()
#ggsave("SGWmin.pdf")

Data.median <- Final.Data.SGW[,-c(1:8)] %>% summarise_all(funs(median))
Data.median.gather <- gather(Data.median)
ggplot(Data.median.gather, aes(value)) + geom_histogram()
#ggsave("SGWmedian.pdf")

Data.mean <- Final.Data.SGW[,-c(1:8)] %>% summarise_all(funs(mean))
Data.mean.gather <- gather(Data.mean)
ggplot(Data.mean.gather, aes(value)) + geom_histogram()
#ggsave("SGWmean.pdf")
```


```{r Correlation, echo=FALSE}

Data.cor <- cor(Final.Data.SGW[,-c(1:8)])
image(flip.over(Data.cor), axes = F,
  frame = T,col = grey(seq(1, 0, length = 256)))
```


```{r trends in data, echo=FALSE}

# Plot 3 days per Node
SGW.3Days <- Final.Data.SGW %>%
  filter(Date.Time > "2018-02-28 00:00:00" & Date.Time < "2018-03-03 00:00:00")
SGW <- ggplot(SGW.3Days, aes(x = Date.Time, y= G19M4C49, color = SGW_ID)) +
  geom_line()
SGW
```

```
#ggplotly(SGW)
#ggsave("SGW_s1u_downlnk_bytes_3.pdf")
head(Final.Data.SGW)
# Plot 7 days whole network
SGW.7Days <- Final.Data.SGW %>%
  filter(Date.Time > "2018-03-03 00:00:00" & Date.Time < "2018-03-10 00:00:00") %>%
  group_by(Date.Time) %>%
  summarise( G19M4C49 = sum(G19M4C49))

SGW <- ggplot(SGW.7Days, aes(x = Date.Time, y= G19M4C49)) +
  geom_line()
SGW

ggsave("SGW_s1u_downlnk_bytes_7_Network.pdf")

# Plot 7 days per Node
SGW.7Days <- Final.Data.SGW %>%
  filter(Date.Time > "2018-03-03 00:00:00" & Date.Time < "2018-03-10 00:00:00")
SGW <- ggplot(SGW.7Days, aes(x = Date.Time, y= G19M4C49, color = SGW_ID)) +
  geom_line()
#ggsave("SGW_s1u_downlnk_bytes_7.pdf")

# Compare Weekdays and Saturdays for the network as a whole

Whole.network <- Final.Data.SGW %>%
  filter(Date.Time > "2018-03-03 00:00:00" &
  Date.Time < "2018-03-10 00:00:00") %>%
  mutate(nTime = as.POSIXct(strftime(Date.Time, format="%H:%M:%S"),
  format="%H:%M:%S")) %>%
  group_by(Day,nTime) %>%
  summarise(G19M4C49 = sum(G19M4C49))

head(Whole.network)
SGW <- ggplot(Whole.network, aes(x = nTime, y= G19M4C49, color = Day)) +
  geom_line()
SGW
#ggplotly(SGW)

# Dip on 3rd at 09h30 on GGMT04
SGW.1Days <- Final.Data.SGW %>% filter(Date.Time > "2018-03-03 08:00:00"
            & Date.Time < "2018-03-03 10:00:00")
            %>% filter (SGW_ID %in% c("GGMT04", "GGMT03"))
SGW <- ggplot(SGW.1Days, aes(x = Date.Time, y= G19M4C49, color = SGW_ID)) +
  geom_line()
#ggplotly(SGW)
```

B.2 PCA analysis

```
---
title: "PCA"
author: "Jason Salzwedel"
---


# Reading data in from file

```{r Read data in, echo = FALSE}

rm(list=ls())
Data <- read.csv("FinalDataSGW.CSV", header = TRUE, sep = ",")
```

```{r PCA all at once, echo = FALSE}

library(plotly)
library(tidyverse)
library(ggplot2)

# Remove factors
Data.less.Factors <- Data[,-c(1:9)]

# Perform Pricincipal Component Analysis
Data.PCA.Prcomp <- prcomp(Data.less.Factors,scale = TRUE, center = TRUE)
write.csv(Data.PCA.Prcomp$x, file = "Data.PCA.Prcomp.csv",row.names=FALSE)

# Plot the Scree plot
PCAsdev <- cumsum(Data.PCA.Prcomp$sdev)
Cumulative_Variances <- PCAsdev/sum(Data.PCA.Prcomp$sdev)
plot(Cumulative_Variances, type = "lines")

# Calculate the variance exmplaind by the x first PCs
sum(Data.PCA.Prcomp$sdev[1:100])/sum(Data.PCA.Prcomp$sdev)

# Join the PCA results back onto the inital factors
PC1PC3 <- cbind(Data[,c(1:9)],Data.PCA.Prcomp$x[,1:3])
```


```{r Investigate PC all at once, echo = FALSE}
# Export the rotations to investigate what the first few PC describe
Rotations <- Data.PCA$rotation
write.csv(Rotations, file =  "Rotations.csv")

#PC1 Consists mainly of normal local Data traffic PDNs setup
# Mobility and session manangemt S5 traffic (all negative.
#PC2 Consists mainly of QCI1 (Positive) and QCI6,7,8,9 trafic
```
```

```
#PC3 Consists mainly of QCI7(Positive)  and QCI2(negagive) traffic


# Plot various dimensions of the first 2 principal components

PCplot <- ggplot(as.tibble(PC1PC3),
  aes(x = PC1, y = PC2, color = Hour)) +
  geom_point(size = 1, stroke = 0, shape = 16)
ggplotly(PCplot)
ggsave("PCAN All Chour.pdf")

PCplot <- ggplot(as.tibble(PC1PC3),
  aes(x = PC1, y = PC2, color = SGW_ID)) +
  geom_point(size = 1, stroke = 0, shape = 16)
PCplot
ggsave("PCAN All CSGW.pdf")

PC1PC3.SGW6 <- PC1PC3 %>%
  filter(SGW_ID %in%
    c("GGCF01","GGCT03","GGCT04","GGDM01","GGDN01","GGDN02"))
PCplot <- ggplot(as.tibble(PC1PC3.SGW6),
aes(x = PC2, y = PC3, Z = PC1, color = Hour)) +
  geom_point(size = 1, stroke = 0, shape = 16) +
  facet_grid(SGW_ID ~ Day)
PCplot
ggsave("PCANAllChourFSD.pdf")

PC1PC3.SGW12 <- PC1PC3 %>%
  filter(SGW_ID %in%
    c("GGJF01","GGMT03","GGMT04","GGPR01","GGPS03","GGPS04"))
PCplot <- ggplot(as.tibble(PC1PC3.SGW12),
  aes(x = PC2, y = PC3, color = Hour)) +
  geom_point(size = 1, stroke = 0, shape = 16) +
  facet_grid(SGW_ID ~ Day)
PCplot
ggsave("PCANAllChourFSDb.pdf")
```
```


#Compare to PCA of all data together with PCA per individual SGW.

```{r PCA all at once, echo = FALSE}

#Plot one SGW using the full original PCA data

PC1PC3.GGJF01 <- cbind(Data[,c(1:9)],Data.PCA$x[,1:3]) %>%
filter(SGW_ID == "GGJF01")
PCplot <- ggplot(as.tibble(PC1PC3.GGJF01),
  aes(x = PC1, y = PC2, color = Hour)) +
  geom_point(size = 1, stroke = 0, shape = 16)
```
```

```
PCplot


PCplot <- ggplot(as.tibble(PC1PC3.GGJF01),
  aes(x = PC1, y = PC2, color = Hour)) +
  geom_point(size = 1, stroke = 0, shape = 16) +
  facet_grid(Hour ~ Day)
PCplot

ggsave("PCAN GGJF01 Chour FHD.pdf")
#ggplotly(PCplot)
```


````
```{r PCA On GGJF01, echo = FALSE}

# Select only the data for GGJF01 and perform the PCA on that data.

GGJF01 <- Data %>% filter(SGW_ID == "GGJF01")

#remove Variables that are always zero for GGJF01
zero.index.GGJF01 <- which(colSums(unname(GGJF01)[,-c(1:9)])==0)
NewGGJF01 <- GGJF01[,-((zero.index.GGJF01)+9)]

Data.less.Factors.GGJF01 <- NewGGJF01[,-c(1:9)]
Data.PCA.GGJF01 <- prcomp(Data.less.Factors.GGJF01,scale = TRUE)

screeplot(Data.PCA.GGJF01)
NewGGJF01PC <- cbind(NewGGJF01[,c(1:9)],Data.PCA.GGJF01$x[,1:3])
PCplot2 <- ggplot(as.tibble(NewGGJF01PC),
  aes(x = -PC1, y = -PC2, color = Hour)) +
  geom_jitter(size = 1, stroke = 0, shape = 16) +
  facet_grid(Hour ~Day)
PCplot2

ggsave("PCAS GGJF01 Chour FHD.pdf")
ggplotly(PCplot2)
```


```{r Kernal PCA all at once, echo = FALSE}
library(kernlab)

Data.scaled <- scale(Data.less.Factors, center = TRUE, scale = TRUE)
Data.scaled <- as.data.frame(Data.scaled)
Data.PCA <- kpca(~.,data=Data.scaled, kernel="rbfdot",
                 kpar=list(sigma=0.2),features=100)


plot(Data.PCA@eig)
````
````

```
sum(100*Data.PCA@eig[1:50]/ sum(Data.PCA@eig))


#print the principal component vectors
rot <- rotated(Data.PCA)
Data.PCA
#plot the data projection on the components
plot(rot[,2:3],col=as.integer(Data[1:1000,9]),
     xlab="1st Principal Component",ylab="2nd Principal Component")
```


```{r KNN, echo = FALSE}
# KNN part of "Class"
# KNN part of "FNN"
library(FNN)

# Calculate the KNN weight from 1 to 50 neighbors
Data.KNN <- knn.dist(Data.PCA.Prcomp$x[,1:100], k = 50)

# Join the results to the factors
Data.KNN.PC1.PC3 <- cbind (Data[,c(1:9)], Data.KNN )

# Investigate the distribution of the weights with K being 10,
# 20, 30, 40 and 50
ggplot(Data.KNN.PC1.PC3, aes(Data.KNN.PC1.PC3[59])) + geom_histogram()
ggplot(Data.KNN.PC1.PC3, aes(Data.KNN.PC1.PC3[49])) + geom_histogram()
ggplot(Data.KNN.PC1.PC3, aes(Data.KNN.PC1.PC3[39])) + geom_histogram()
ggplot(Data.KNN.PC1.PC3, aes(Data.KNN.PC1.PC3[29])) + geom_histogram()
ggplot(Data.KNN.PC1.PC3, aes(Data.KNN.PC1.PC3[19])) + geom_histogram()
```

```{r KNN as point anomalies, echo = FALSE}
# As discovered earlier, the anomalies need to be considered in context.
#To move from contextual anomaly detection to point anomaly detection,
#the data set will be broken up into subsets. Various combinations of
#subsets will be investigated.

Data.PCA100 <- cbind(Data[,c(1:9)],Data.PCA.Prcomp$x[,1:100])

# A nested for loop calculated the KNN per Node/Time subset and then adds
# the results to the orignial dataset
Nodes <- unique(Data.PCA100$SGW_ID)
Times <- unique(Data.PCA100$Time)
str(Data.PCA100)
KNNData.index <- c(1:121)
for (Node in  Nodes){
  print(Node)
  for (TOD in Times){
   print(TOD)
```

```
    KNNsubset <- Data.PCA100 %>% filter(SGW_ID == Node & Time ==TOD) %>%
      select( c(10:109)) %>% knn.dist(k =12)
    KNNindex  <- Data.PCA100 %>% filter(SGW_ID == Node & Time ==TOD) %>%
      select(X)
    KNNData.index <- rbind(KNNData.index, cbind (KNNindex, KNNsubset))
   }
}

KNN.Data.Node.Time <- inner_join(Data.PCA100, KNNData.index, by = "X")
names(KNN.Data.Node.Time)[110:121] <- c("KNN1","KNN2","KNN3","KNN4",
  "KNN5", "KNN6","KNN7","KNN8","KNN9","KNN10","KNN11","KNN12")
KNN.Data.Node.Time <- KNN.Data.Node.Time %>%
mutate( Weight12 = (KNN1+KNN2+KNN3+KNN4+KNN5+KNN6+KNN7+
  KNN8+KNN9+KNN10+KNN11+KNN12)/12)
Outliers <- ggplot(KNN.Data.Node.Time, aes(Weight12)) +  geom_histogram()
Outliers

KNN.Data.Node.Time %>% filter(KNN.Data.Node.Time$Weight12 > 200)
```

B.3 K Nearest Neighbors

```
---
title: "SGW KNN"
author: "Jason Salzwedel"
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
rm(list=ls())
Data <- read.csv("FinalDataSGW.CSV", header = TRUE, sep = ",")
Data.PCA.Prcomp <- read.csv("Data.PCA.Prcomp.csv", header = TRUE,
  sep = ",")
```

```{r KNN on combined data set, echo = Falsesd}
# KNN part of "Class"
# KNN part of "FNN"
library(FNN)
library(ggplot2)
library(tidyverse)

# Calculate the KNN weight from 1 to 50 neighbors
Data.KNN <- knn.dist(Data.PCA.Prcomp[,1:100], k = 50)

# Join the results to the factors
Data.PCA.KNN <- cbind (Data[,c(1:9)], Data.KNN )

# Investigate the distribution of the distance to Kth with K being
```
```

```
# 10, 20, 30, 40 and 50
ggplot(Data.PCA.KNN, aes(Data.PCA.KNN[59])) + geom_histogram() +
    scale_y_sqrt() + labs(x ="Kth = 50" , y = "")
ggplot(Data.PCA.KNN, aes(Data.PCA.KNN[49])) + geom_histogram() +
    scale_y_sqrt() + labs(x ="Kth = 40" , y = "")
ggplot(Data.PCA.KNN, aes(Data.PCA.KNN[39])) + geom_histogram() +
    scale_y_sqrt() + labs(x ="Kth = 30" , y = "")
ggplot(Data.PCA.KNN, aes(Data.PCA.KNN[29])) + geom_histogram() +
    scale_y_sqrt() + labs(x ="Kth = 20" , y = "")
ggplot(Data.PCA.KNN, aes(Data.PCA.KNN[19])) + geom_histogram() +
    scale_y_sqrt() + labs(x ="Kth = 10" , y = "")

max(Data.PCA.KNN[59])
names(Data.PCA.KNN)[60:61]
# Rename the columes from number to char
names(Data.PCA.KNN)[10:59] <- c("KNN1","KNN2","KNN3","KNN4","KNN5",
  "KNN6","KNN7","KNN8","KNN9","KNN10", "KNN11","KNN12","KNN13","KNN14",
  "KNN15","KNN16","KNN17","KNN18","KNN19","KNN20", "KNN21","KNN22",
  "KNN23", "KNN24","KNN25","KNN26","KNN27","KNN28","KNN29","KNN30",
  "KNN31","KNN32","KNN33","KNN34","KNN35","KNN36","KNN37","KNN38",
  "KNN39","KNN40","KNN41","KNN42","KNN43","KNN44","KNN45","KNN46",
  "KNN47","KNN48","KNN49","KNN50")


# Calculate the KNN weights
Data.PCA.KNN <- Data.PCA.KNN %>%
mutate( Weight10 = (KNN1 +KNN2 +KNN3 +KNN4 +KNN5 +KNN6 +KNN7 +KNN8
  +KNN9 +KNN10)/10)
Data.PCA.KNN <- Data.PCA.KNN %>%
mutate( Weight20 = ((Weight10*10) +KNN11+KNN12+KNN13+KNN14+KNN15+KNN16+
  KNN17+KNN18+KNN19+KNN20)/20)
Data.PCA.KNN <- Data.PCA.KNN %>%
mutate( Weight30 = ((Weight20*20) +KNN21+KNN22+KNN23+KNN24+KNN25+KNN26+
  KNN27+KNN28+KNN29+KNN30)/30)
Data.PCA.KNN <- Data.PCA.KNN %>%
mutate( Weight40 = ((Weight30*30) +KNN31+KNN32+KNN33+KNN34+KNN35+KNN36+
  KNN37+KNN38+KNN39+KNN40)/40)
Data.PCA.KNN <- Data.PCA.KNN %>%
mutate( Weight50 = ((Weight40*40) +KNN41+KNN42+KNN43+KNN44+KNN45+KNN46+
  KNN47+KNN48+KNN49+KNN50)/50)


write.csv(Data.PCA.KNN[,-c(2:9)], file ="KNN.csv")
# Identify the top


# Investigate the distribution of the distance to Kth with K being
# 10, 20, 30, 40 and 50
ggplot(Data.PCA.KNN, aes(Data.PCA.KNN$Weight50 )) + geom_histogram() +
  scale_y_sqrt() + labs(x ="Weight with K = 50" , y = "")
```

```
ggplot(Data.PCA.KNN, aes(Data.PCA.KNN$Weight40 )) + geom_histogram() +
  scale_y_sqrt() + labs(x ="Weight with K = 40" , y = "")
ggplot(Data.PCA.KNN, aes(Data.PCA.KNN$Weight30 )) + geom_histogram() +
  scale_y_sqrt() + labs(x ="Weight with K = 30" , y = "")
ggplot(Data.PCA.KNN, aes(Data.PCA.KNN$Weight20 )) + geom_histogram() +
  scale_y_sqrt() + labs(x ="Weight with K = 20" , y = "")
ggplot(Data.PCA.KNN, aes(Data.PCA.KNN$Weight10 )) + geom_histogram() +
  scale_y_sqrt() + labs(x ="Weight with K = 10" , y = "")
```

```
```{r Investigate global anomalies, echo = Falsesd}

# Add rank to identify top anomalies

Data.PCA.KNN <- Data.PCA.KNN %>%
  mutate(rankKNN50 = dense_rank(desc(KNN50)))
Data.PCA.KNN <- Data.PCA.KNN %>%
  mutate(rankKNN40 = dense_rank(desc(KNN40)))
Data.PCA.KNN <- Data.PCA.KNN %>%
  mutate(rankKNN30 = dense_rank(desc(KNN30)))
Data.PCA.KNN <- Data.PCA.KNN %>%
  mutate(rankKNN20 = dense_rank(desc(KNN20)))
Data.PCA.KNN <- Data.PCA.KNN %>%
  mutate(rankKNN10 = dense_rank(desc(KNN10)))

Data.PCA.KNN <- Data.PCA.KNN %>%
  mutate(rankWeight50 = dense_rank(desc(Weight50)))
Data.PCA.KNN <- Data.PCA.KNN %>%
  mutate(rankWeight40 = dense_rank(desc(Weight40)))
Data.PCA.KNN <- Data.PCA.KNN %>%
  mutate(rankWeight30 = dense_rank(desc(Weight30)))
Data.PCA.KNN <- Data.PCA.KNN %>%
  mutate(rankWeight20 = dense_rank(desc(Weight20)))
Data.PCA.KNN <- Data.PCA.KNN %>%
  mutate(rankWeight10 = dense_rank(desc(Weight10)))

Outlier100 <- Data.PCA.KNN %>% arrange(rankKNN50) %>% select(X)
Outlier100 <- cbind(Outlier100,Data.PCA.KNN %>%
  arrange(rankKNN40) %>% select(X) )
Outlier100 <- cbind(Outlier100,Data.PCA.KNN %>%
  arrange(rankKNN30) %>% select(X) )
Outlier100 <- cbind(Outlier100,Data.PCA.KNN %>%
  arrange(rankKNN20) %>% select(X) )
Outlier100 <- cbind(Outlier100,Data.PCA.KNN %>%
  arrange(rankKNN10) %>% select(X) )
Outlier100 <- cbind(Outlier100,Data.PCA.KNN %>%
  arrange(rankWeight50) %>% select(X) )
Outlier100 <- cbind(Outlier100,Data.PCA.KNN %>%
  arrange(rankWeight40) %>% select(X) )
```
```

```
Outlier100 <- cbind(Outlier100,Data.PCA.KNN %>%
  arrange(rankWeight30) %>% select(X) )
Outlier100 <- cbind(Outlier100,Data.PCA.KNN %>%
  arrange(rankWeight20) %>% select(X) )
Outlier100 <- cbind(Outlier100,Data.PCA.KNN %>%
  arrange(rankWeight10) %>% select(X) )

names(Outlier100) <- c("OutlierRankKNN50",
                               "OutlierRankKNN40",
                               "OutlierRankKNN30",
                               "OutlierRankKNN20",
                               "OutlierRankKNN10",
                               "OutlierWeightKNN50",
                               "OutlierWeightKNN40",
                               "OutlierWeightKNN30",
                               "OutlierWeightKNN20",
                               "OutlierWeightKNN10")


# Create a matrix comparing each of the top 10 KNN approaches
# with each other (how many match)


Top100Outlier <- Outlier100[1:100,]
MatchMatrix <- matrix(c(1:100), ncol = 10)

i = 1
j = 1
for(i in 1:10){
  for(j in 1:10){
    matchmatch <- sum(Top100Outlier[,i] %in% Top100Outlier[,j])
    MatchMatrix[i,j] <- matchmatch
  }
}

MatchMatrix <- as.data.frame(MatchMatrix)

names(MatchMatrix) <- c("50th NN",
                               "40th NN",
                               "30th NN",
                               "20th NN",
                               "10th NN0",
                               "Weight of 50",
                               "Weight of 40",
                               "Weight of 30",
                               "Weight of 20",
                               "Weight of 10")

rownames(MatchMatrix) <- c("50th NN",
                               "40th NN",
```

```
                                "30th NN",
                                "20th NN",
                                "10th NN0",
                                "Weight of 50",
                                "Weight of 40",
                                "Weight of 30",
                                "Weight of 20",
                                "Weight of 10")

length(unique(as.vector(as.matrix(Top100Outlier))))



```

```{r KNN as point anomalies based on Node grouping, echo = Falsesd}
# As discovered earlier, the anomalies need to be considered in context.
# To move from contextual anomaly detection to point anomaly detection,
# the data set will be broken up into subsets. Various combinations of
# subsets will be investigated.

Data.PCA100 <- cbind(Data[,c(1:9)],Data.PCA.Prcomp[,1:100])

# A nested for loop calculated the KNN per Node/Time subset and then
# adds the results to the orignial dataset

Nodes <- unique(Data.PCA100$SGW_ID)

KNNData.index <- c(1:169)
for (Node in Nodes){
    KNNsubset <- Data.PCA100 %>% filter(SGW_ID == Node) %>%
      select( c(10:109)) %>% knn.dist(k =50)
    KNNindex  <- Data.PCA100 %>% filter(SGW_ID == Node) %>%
      select(X)
    KNNData.index <- rbind(KNNData.index, cbind (KNNindex, KNNsubset))
   }



# Join the calculated KNN values onto the original data set
KNN.Data.Node.Time <- inner_join(Data.PCA.KNN, KNNData.index[-1,], by = "X")

# Rename the columes from number to char

names(KNN.Data.Node.Time)[75:124] <- c("NODEKNN1","NODEKNN2", "NODEKNN3",
  "NODEKNN4","NODEKNN5","NODEKNN6","NODEKNN7","NODEKNN8","NODEKNN9",
  "NODEKNN10", "NODEKNN11","NODEKNN12","NODEKNN13","NODEKNN14",
  "NODEKNN15","NODEKNN16","NODEKNN17","NODEKNN18","NODEKNN19","NODEKNN20",
  "NODEKNN21","NODEKNN22","NODEKNN23","NODEKNN24","NODEKNN25","NODEKNN26",
  "NODEKNN27","NODEKNN28","NODEKNN29","NODEKNN30","NODEKNN31","NODEKNN32",
```

```
  "NODEKNN33","NODEKNN34","NODEKNN35","NODEKNN36","NODEKNN37","NODEKNN38",
  "NODEKNN39","NODEKNN40","NODEKNN41","NODEKNN42","NODEKNN43","NODEKNN44",
  "NODEKNN45","NODEKNN46","NODEKNN47","NODEKNN48","NODEKNN49","NODEKNN50")
# Calculate the NODE weight to the first 12 neighbors

# Calculate the KNN weights
KNN.Data.Node.Time <- KNN.Data.Node.Time %>%
mutate( NodeWeight10 = (NODEKNN1 +NODEKNN2 +NODEKNN3 +NODEKNN4 +NODEKNN5
  +NODEKNN6 +NODEKNN7 +NODEKNN8 +NODEKNN9 +NODEKNN10)/10)
KNN.Data.Node.Time <- KNN.Data.Node.Time %>%
  mutate( NodeWeight20 = ((NodeWeight10*10) +NODEKNN11+NODEKNN12+NODEKNN13
  +NODEKNN14+NODEKNN15+NODEKNN16+NODEKNN17+NODEKNN18+NODEKNN19+NODEKNN20)/20)
KNN.Data.Node.Time <- KNN.Data.Node.Time %>%
  mutate( NodeWeight30 = ((NodeWeight20*20) +NODEKNN21+NODEKNN22+NODEKNN23+
  NODEKNN24+NODEKNN25+NODEKNN26+NODEKNN27+NODEKNN28+NODEKNN29+NODEKNN30)/30)
KNN.Data.Node.Time <- KNN.Data.Node.Time %>%
  mutate( NodeWeight40 = ((NodeWeight30*30) +NODEKNN31+NODEKNN32+NODEKNN33+
  NODEKNN34+NODEKNN35+NODEKNN36+NODEKNN37+NODEKNN38+NODEKNN39+NODEKNN40)/40)
KNN.Data.Node.Time <- KNN.Data.Node.Time %>%
  mutate( NodeWeight50 = ((NodeWeight40*40) +NODEKNN41+NODEKNN42+NODEKNN43+
  NODEKNN44+NODEKNN45+NODEKNN46+NODEKNN47+NODEKNN48+NODEKNN49+NODEKNN50)/50)

# Check that the distance to the total data set is always greater to the subset
# than to the total set
sum (KNN.Data.Node.Time$NODEKNN50 < KNN.Data.Node.Time$KNN50)

# Add the rankings to the dataset

KNN.Data.Node.Time <- KNN.Data.Node.Time %>%
  mutate(rankNodeKNN50 = dense_rank(desc(NODEKNN50)))
KNN.Data.Node.Time <- KNN.Data.Node.Time %>%
  mutate(rankNodeKNN40 = dense_rank(desc(NODEKNN40)))
KNN.Data.Node.Time <- KNN.Data.Node.Time %>%
  mutate(rankNodeKNN30 = dense_rank(desc(NODEKNN30)))
KNN.Data.Node.Time <- KNN.Data.Node.Time %>%
  mutate(rankNodeKNN20 = dense_rank(desc(NODEKNN20)))
KNN.Data.Node.Time <- KNN.Data.Node.Time %>%
  mutate(rankNodeKNN10 = dense_rank(desc(NODEKNN10)))

KNN.Data.Node.Time <- KNN.Data.Node.Time %>%
  mutate(rankNodeWeight50 = dense_rank(desc(NodeWeight50)))
KNN.Data.Node.Time <- KNN.Data.Node.Time %>%
  mutate(rankNodeWeight40 = dense_rank(desc(NodeWeight40)))
KNN.Data.Node.Time <- KNN.Data.Node.Time %>%
  mutate(rankNodeWeight30 = dense_rank(desc(NodeWeight30)))
KNN.Data.Node.Time <- KNN.Data.Node.Time %>%
  mutate(rankNodeWeight20 = dense_rank(desc(NodeWeight20)))
KNN.Data.Node.Time <- KNN.Data.Node.Time %>%
  mutate(rankNodeWeight10 = dense_rank(desc(NodeWeight10)))
```

```
# Extend the outlier list with the Node based Data

Outlier100 <- cbind(Outlier100,KNN.Data.Node.Time %>%
  arrange(rankNodeKNN50) %>% select(X) )
Outlier100 <- cbind(Outlier100,KNN.Data.Node.Time %>%
  arrange(rankNodeKNN40) %>% select(X) )
Outlier100 <- cbind(Outlier100,KNN.Data.Node.Time %>%
  arrange(rankNodeKNN30) %>% select(X) )
Outlier100 <- cbind(Outlier100,KNN.Data.Node.Time %>%
  arrange(rankNodeKNN20) %>% select(X) )
Outlier100 <- cbind(Outlier100,KNN.Data.Node.Time %>%
  arrange(rankNodeKNN10) %>% select(X) )
Outlier100 <- cbind(Outlier100,KNN.Data.Node.Time %>%
  arrange(rankNodeWeight50) %>% select(X) )
Outlier100 <- cbind(Outlier100,KNN.Data.Node.Time %>%
  arrange(rankNodeWeight40) %>% select(X) )
Outlier100 <- cbind(Outlier100,KNN.Data.Node.Time %>%
  arrange(rankNodeWeight30) %>% select(X) )
Outlier100 <- cbind(Outlier100,KNN.Data.Node.Time %>%
  arrange(rankNodeWeight20) %>% select(X) )
Outlier100 <- cbind(Outlier100,KNN.Data.Node.Time %>%
  arrange(rankNodeWeight10) %>% select(X) )

names(Outlier100) <- c("OutlierRankKNN50",
                               "OutlierRankKNN40",
                               "OutlierRankKNN30",
                               "OutlierRankKNN20",
                               "OutlierRankKNN10",
                               "OutlierWeightKNN50",
                               "OutlierWeightKNN40",
                               "OutlierWeightKNN30",
                               "OutlierWeightKNN20",
                               "OutlierWeightKNN10",
                               "OutlierRankNODEKNN50",
                               "OutlierRankNODEKNN40",
                               "OutlierRankNODEKNN30",
                               "OutlierRankNODEKNN20",
                               "OutlierRankNODEKNN10",
                               "OutlierNODEWeightKNN50",
                               "OutlierNODEWeightKNN40",
                               "OutlierNODEWeightKNN30",
                               "OutlierNODEWeightKNN20",
                               "OutlierNODEWeightKNN10")

# Create a matrix comparing each of the top 10 KNN approaches
#with each other (how many match)
```

```
Top100Outlier <- Outlier100[1:100,]
MatchMatrixNode <- matrix(c(1:100), ncol = 10)

i = 1
j = 1
for(i in 1:10){
  for(j in 1:10){
    matchmatch <- sum(Top100Outlier[,i+10] %in% Top100Outlier[,j+10])
    MatchMatrixNode[i,j] <- matchmatch

  }

}

MatchMatrixNode <- as.data.frame(MatchMatrixNode)

names(MatchMatrixNode) <- c("50th NN",
                            "40th NN",
                            "30th NN",
                            "20th NN",
                            "10th NN0",
                            "Weight of 50",
                            "Weight of 40",
                            "Weight of 30",
                            "Weight of 20",
                            "Weight of 10")

rownames(MatchMatrixNode) <- c("50th NN",
                            "40th NN",
                            "30th NN",
                            "20th NN",
                            "10th NN0",
                            "Weight of 50",
                            "Weight of 40",
                            "Weight of 30",
                            "Weight of 20",
                            "Weight of 10")


# How many outliers are identified by the 10 different KNN approaches
# based on the Node grouping distances

length(unique(as.vector(as.matrix(Top100Outlier[11:20]))))


# How Many are the same?
sum(unique(as.vector(as.matrix(Top100Outlier[11:20]))) %in%
  unique(as.vector(as.matrix(Top100Outlier[1:10]))))
# Answer 110 of the 114 identified by the Node based approach are
# in the original 124 identified by the approach looking at all the data.
```

```
# What are the these 4 new points and what are the rankings?
NodeOutliers.NewFlag <- cbind(unique(as.vector(as.matrix(Top100Outlier[11:20])))
  %in% unique(as.vector(as.matrix(Top100Outlier[1:10]))),
    unique(as.vector(as.matrix(Top100Outlier[11:20]))))

New.Outliers.Node <- as.data.frame(NodeOutliers.NewFlag) %>%
  filter(V1 ==0)

# how do the new node based weights/distances compare to the old
# previous total view?
a <- KNN.Data.Node.Time %>% filter(X %in% New.Outliers.Node$V2 )

# Create a table of new observations and their new and old rankings
table.of.new <- matrix(c(1:12), ncol= 3)
for (i in c(1:4)){
  table.of.new[i,1] <- a[i,1]
  table.of.new[i,2] <- mean(as.matrix(a[i,65:74]))
  table.of.new[i,3] <- mean(as.matrix(a[i,130:139]))
}



# Which points were marked as outliers in the total view that are not in
# the node view?

sum(!(unique(as.vector(as.matrix(Top100Outlier[1:10]))) %in%
  unique(as.vector(as.matrix(Top100Outlier[11:20])))))
# 14

# Which are those 14 ?

#Create a matrix with the observation number and a flag indicating it is
# not in the node group.
OriginalOutliers.missedFlag <-
  cbind  ( !(unique(as.vector(as.matrix(Top100Outlier[1:10]))) %in%
  unique(as.vector(as.matrix(Top100Outlier[11:20])))),
  unique(as.vector(as.matrix(Top100Outlier[1:10]))))

Missed.Outliers.Node <- as.data.frame(OriginalOutliers.missedFlag) %>%
  filter(V1 ==1)

# how do the missed original based weights/distances compare to new
# node based view?
b <- KNN.Data.Node.Time %>% filter(X %in% Missed.Outliers.Node$V2 )

# Create a table showing the observations and their average KNN
# distances/weights for the node based on total based
```

```
table.of.lost <- matrix(c(1:42), ncol= 3)
for (i in c(1:14)){
  table.of.lost[i,1] <- b[i,1]
  table.of.lost[i,2] <- mean(as.matrix(b[i,65:74]))
  table.of.lost[i,3] <- mean(as.matrix(b[i,130:139]))
}

table.of.lost <- as.data.frame(table.of.lost)
names(table.of.lost) <- c("Observation", "Original KNN","Node based KNN" )

write.csv(KNN.Data.Node.Time[,c(1,75:129)], file = "Node KNN.csv")
```

````
```{r KNN as point anomalies based on Node and Time, echo = Falsesd}

# As discovered earlier, the anomalies need to be considered in context.
# To move from contextual anomaly detection to point anomaly detection,
# the data set will be broken up into subsets. Various combinations of
# subsets will be investigated. Here we group based on Node and time
# of day. This leaves 12 observations per group

Data.PCA100 <- cbind(Data[,c(1:9)],Data.PCA.Prcomp[,1:100])

# A nested for loop calculated the KNN per Node/Time subset and then
# adds the results to the orignial dataset

Nodes <- unique(Data.PCA100$SGW_ID)
Times <- unique(Data.PCA100$Time)

KNNData.index <- c(1:119)
for (Node in  Nodes){
  for (TOD in Times){
    KNNsubset <- Data.PCA100 %>% filter(SGW_ID == Node & Time ==TOD) %>%
    select( c(10:109)) %>% knn.dist(k =10)
    KNNindex  <- Data.PCA100 %>% filter(SGW_ID == Node & Time ==TOD) %>%
    select(X)
    KNNData.index <- rbind(KNNData.index, cbind (KNNindex, KNNsubset))
   }
}


# Join the calculated KNN values onto the original data set
KNN.Data.Node.Time <-
  inner_join(KNN.Data.Node.Time, KNNData.index[-1,], by = "X")

# Rename the columes from number to char
names(KNN.Data.Node.Time)[140:149] <- c("NODETIMEKNN1","NODETIMEKNN2",
  "NODETIMEKNN3","NODETIMEKNN4","NODETIMEKNN5","NODETIMEKNN6",
    "NODETIMEKNN7","NODETIMEKNN8","NODETIMEKNN9","NODETIMEKNN10")
````
````

```
# Calculate the KNN weight to the first 12 neighbors
KNN.Data.Node.Time <- KNN.Data.Node.Time %>%
 mutate( NodeTimeWeight2 = (NODETIMEKNN1 + NODETIMEKNN2) /2)
KNN.Data.Node.Time <- KNN.Data.Node.Time %>%
  mutate( NodeTimeWeight4 = ((NodeTimeWeight2*2) +NODETIMEKNN3 +
    NODETIMEKNN4) /4)
KNN.Data.Node.Time <- KNN.Data.Node.Time %>%
  mutate( NodeTimeWeight6 = ((NodeTimeWeight4*4) +NODETIMEKNN5 +
    NODETIMEKNN6) /6)
KNN.Data.Node.Time <- KNN.Data.Node.Time %>%
  mutate( NodeTimeWeight8 = ((NodeTimeWeight6*6) +NODETIMEKNN7 +
    NODETIMEKNN8) /8)
KNN.Data.Node.Time <- KNN.Data.Node.Time %>%
  mutate( NodeTimeWeight10 =((NodeTimeWeight8*8) +NODETIMEKNN9 +
    NODETIMEKNN10)/10)

# Check that the distance to the total data set is always greater
# to the subset than to the total set
sum (KNN.Data.Node.Time$NODETIMEKNN10 < KNN.Data.Node.Time$NODEKNN10)

# Add the rankings to the dataset

KNN.Data.Node.Time <- KNN.Data.Node.Time %>%
  mutate(rankNodeTimeKNN10 = dense_rank(desc(NODETIMEKNN10)))
KNN.Data.Node.Time <- KNN.Data.Node.Time %>%
  mutate(rankNodeTimeKNN8  = dense_rank(desc(NODETIMEKNN8)))
KNN.Data.Node.Time <- KNN.Data.Node.Time %>%
  mutate(rankNodeTimeKNN6  = dense_rank(desc(NODETIMEKNN6)))
KNN.Data.Node.Time <- KNN.Data.Node.Time %>%
  mutate(rankNodeTimeKNN4  = dense_rank(desc(NODETIMEKNN4)))
KNN.Data.Node.Time <- KNN.Data.Node.Time %>%
  mutate(rankNodeTimeKNN2  = dense_rank(desc(NODETIMEKNN2)))

KNN.Data.Node.Time <- KNN.Data.Node.Time %>%
  mutate(rankNodeTimeWeight10 = dense_rank(desc(NodeTimeWeight10)))
KNN.Data.Node.Time <- KNN.Data.Node.Time %>%
  mutate(rankNodeTimeWeight8 =  dense_rank(desc(NodeTimeWeight8)))
KNN.Data.Node.Time <- KNN.Data.Node.Time %>%
  mutate(rankNodeTimeWeight6 =  dense_rank(desc(NodeTimeWeight6)))
KNN.Data.Node.Time <- KNN.Data.Node.Time %>%
  mutate(rankNodeTimeWeight4 =  dense_rank(desc(NodeTimeWeight4)))
KNN.Data.Node.Time <- KNN.Data.Node.Time %>%
  mutate(rankNodeTimeWeight2 =  dense_rank(desc(NodeTimeWeight2)))

# Extend the outlier list with the Node based Data

Outlier100 <- cbind(Outlier100,KNN.Data.Node.Time %>%
  arrange(rankNodeTimeKNN10) %>% select(X) )
Outlier100 <- cbind(Outlier100,KNN.Data.Node.Time %>%
```

```
  arrange(rankNodeTimeKNN8) %>% select(X) )
Outlier100 <- cbind(Outlier100,KNN.Data.Node.Time %>%
  arrange(rankNodeTimeKNN6) %>% select(X) )
Outlier100 <- cbind(Outlier100,KNN.Data.Node.Time %>%
  arrange(rankNodeTimeKNN4) %>% select(X) )
Outlier100 <- cbind(Outlier100,KNN.Data.Node.Time %>%
  arrange(rankNodeTimeKNN2) %>% select(X) )
Outlier100 <- cbind(Outlier100,KNN.Data.Node.Time %>%
  arrange(rankNodeTimeWeight10) %>% select(X) )
Outlier100 <- cbind(Outlier100,KNN.Data.Node.Time %>%
  arrange(rankNodeTimeWeight8) %>% select(X) )
Outlier100 <- cbind(Outlier100,KNN.Data.Node.Time %>%
  arrange(rankNodeTimeWeight6) %>% select(X) )
Outlier100 <- cbind(Outlier100,KNN.Data.Node.Time %>%
  arrange(rankNodeTimeWeight4) %>% select(X) )
Outlier100 <- cbind(Outlier100,KNN.Data.Node.Time %>%
  arrange(rankNodeTimeWeight2) %>% select(X) )

names(Outlier100) <- c("OutlierRankKNN50",
                               "OutlierRankKNN40",
                               "OutlierRankKNN30",
                               "OutlierRankKNN20",
                               "OutlierRankKNN10",
                               "OutlierWeightKNN50",
                               "OutlierWeightKNN40",
                               "OutlierWeightKNN30",
                               "OutlierWeightKNN20",
                               "OutlierWeightKNN10",
                               "OutlierRankNODEKNN50",
                               "OutlierRankNODEKNN40",
                               "OutlierRankNODEKNN30",
                               "OutlierRankNODEKNN20",
                               "OutlierRankNODEKNN10",
                               "OutlierNODEWeightKNN50",
                               "OutlierNODEWeightKNN40",
                               "OutlierNODEWeightKNN30",
                               "OutlierNODEWeightKNN20",
                               "OutlierNODEWeightKNN10",
                               "OutlierRankNODETIMEKNN10",
                               "OutlierRankNODETIMEKNN8",
                               "OutlierRankNODETIMEKNN6",
                               "OutlierRankNODETIMEKNN4",
                               "OutlierRankNODETIMEKNN2",
                               "OutlierNODETIMEWeightKNN10",
                               "OutlierNODETIMEWeightKNN8",
                               "OutlierNODETIMEWeightKNN6",
                               "OutlierNODETIMEWeightKNN4",
                               "OutlierNODETIMEWeightKNN2")

# Create a matrix comparing each of the top 10 KNN approaches
```

```
# with each other (how many match)



Top100Outlier <- Outlier100[1:100,]
MatchMatrixNodeTime <- matrix(c(1:100), ncol = 10)

i = 1
j = 1
for(i in 1:10){
  for(j in 1:10){
    matchmatch <- sum(Top100Outlier[,i+20] %in% Top100Outlier[,j+20])
    MatchMatrixNodeTime[i,j] <- matchmatch

  }

}

MatchMatrixNodeTime <- as.data.frame(MatchMatrixNodeTime)

names(MatchMatrixNodeTime) <- c("50th NN",
                                "40th NN",
                                "30th NN",
                                "20th NN",
                                "10th NN0",
                                "Weight of 50",
                                "Weight of 40",
                                "Weight of 30",
                                "Weight of 20",
                                "Weight of 10")

rownames(MatchMatrixNodeTime) <- c("50th NN",
                                "40th NN",
                                "30th NN",
                                "20th NN",
                                "10th NN0",
                                "Weight of 50",
                                "Weight of 40",
                                "Weight of 30",
                                "Weight of 20",
                                "Weight of 10")

# How many outliers are identified by the 10 different KNN
#  approaches based on the Node/Time grouping distances

length(unique(as.vector(as.matrix(Top100Outlier[21:30]))))
#104

# How Many are the same as in the original?
sum(unique(as.vector(as.matrix(Top100Outlier[21:30]))) %in%
```

```
  unique(as.vector(as.matrix(Top100Outlier[ 1:10]))))
#101

# How Many are the same as in the Node based approach?
sum(unique(as.vector(as.matrix(Top100Outlier[21:30]))) %in%
  unique(as.vector(as.matrix(Top100Outlier[11:20]))))
#100

# What are the these 4 new points not seen in the original
# rankings and what are the rankings?
NodeTimeOutliers.NewFlag.vsO <-
  cbind(unique(as.vector(as.matrix(Top100Outlier[21:30]))) %in%
    unique(as.vector(as.matrix(Top100Outlier[1:10]))),
      unique(as.vector(as.matrix(Top100Outlier[21:30]))))

New.Outliers.Node.Time <-
  as.data.frame(NodeTimeOutliers.NewFlag.vsO) %>% filter(V1 ==0)

# how do the new node/time based weights/distances compare to
# the old previous total view?
c <- KNN.Data.Node.Time %>% filter(X %in% New.Outliers.Node.Time$V2 )

# Create a table of new observations and their new and old rankings
table.of.new.node.time <- matrix(c(1:12), ncol= 4)
for (i in c(1:3)){
  table.of.new.node.time[i,1] <- c[i,1]
  table.of.new.node.time[i,2] <- mean(as.matrix(c[i,65:74]))
  table.of.new.node.time[i,3] <- mean(as.matrix(c[i,130:139]))
  table.of.new.node.time[i,4] <- mean(as.matrix(c[i,155:164]))
}

# What are the these 3 new points not seen in the Node based
#  rankings and what are the rankings?
NodeTimeOutliers.NewFlag.vsN <-
  cbind(unique(as.vector(as.matrix(Top100Outlier[21:30]))) %in%
    unique(as.vector(as.matrix(Top100Outlier[11:20]))),
      unique(as.vector(as.matrix(Top100Outlier[21:30]))))

New.Outliers.Node.Time.Vn <-
  as.data.frame(NodeTimeOutliers.NewFlag.vsN) %>% filter(V1 ==0)

# How do the new node/time based weights/distances compare to the old
# previous total view?
d <- KNN.Data.Node.Time %>% filter(X %in% New.Outliers.Node.Time.Vn$V2)

# Create a table of new observations and their new and old rankings
table.of.new.node.timevN <- matrix(c(1:16), ncol= 4)
for (i in c(1:4)){
  table.of.new.node.timevN[i,1] <- d[i,1]
  table.of.new.node.timevN[i,2] <- mean(as.matrix(d[i,65:74]))
```

```
  table.of.new.node.timevN[i,3] <- mean(as.matrix(d[i,130:139]))
  table.of.new.node.timevN[i,4] <- mean(as.matrix(d[i,155:164]))
}


# Which points were marked as outliers in the total view that are not
# in the node view?

sum(!(unique(as.vector(as.matrix(Top100Outlier[1:20]))) %in%
  unique(as.vector(as.matrix(Top100Outlier[21:30])))))
# 26

# Which are those 26 ?

#Create a matrix with the observation number and a flag indicating
# it is not in the node group.
OriginalOutliers.missedFlag <-
  cbind  ( !(unique(as.vector(as.matrix(Top100Outlier[1:20]))) %in%
    unique(as.vector(as.matrix(Top100Outlier[21:30])))),
      unique(as.vector(as.matrix(Top100Outlier[1:20]))))

Missed.Outliers.Node <- as.data.frame(OriginalOutliers.missedFlag) %>%
  filter(V1 ==1)

# How do the missed original based weights/distances compare to new
# node based view?

e <- KNN.Data.Node.Time %>% filter(X %in% Missed.Outliers.Node$V2 )

# Create a table showing the observations and their average KNN
# distances/weights for the node based on total based

table.of.lost <- matrix(c(1:104), ncol= 4)
for (i in c(1:26)){
  table.of.lost[i,1] <- e[i,1]
  table.of.lost[i,2] <- mean(as.matrix(e[i,65:74]))
  table.of.lost[i,3] <- mean(as.matrix(e[i,130:139]))
  table.of.lost[i,4] <- mean(as.matrix(e[i,155:164]))
}

table.of.lost <- as.data.frame(table.of.lost)
names(table.of.lost) <- c("Observation", "Original KNN","Node based KNN" )

write.csv(KNN.Data.Node.Time[,c(1,140:154)],file = "Node Time KNN.csv")
```

B.4 One-Class Support Vector Machines

```
---
title: "SGW OCSVM"
```

```
author: "Jason Salzwedel"
---

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
rm(list=ls())
# Read in data
Data <- read.csv("FinalDataSGW.CSV", header = TRUE, sep = ",")
Data.PCA.Prcomp <- read.csv("Data.PCA.Prcomp.csv", header = TRUE, sep = ",")
```


```{r Perform OCSVM analysis with Varying parameters}

library (e1071)
library (tidyverse)
library (ggplot2)

#Prepare the parameters for the grid search

#search 1
my.nu    <- c(0.01, 0.05, 0.1, 0.5, 1)
my.gamma <- c(0.01, 0.05, 0.1, 0.5, 1)
#search 2
my.nu    <- c(0.05,0.1,0.15)
my.gamma <- c(0.001,0.005,0.01,0.02)
#search 3
my.nu    <- c(0.13,0.18,0.2)
my.gamma <- c(0.0007,0.002,0.0055)

# Create a dataframe to hold the results
oneclass.d.values <- rep(0,16070)
oneclass.d.values <- as.data.frame(oneclass.d.values)

#create the SVM models and results through a nested look which
#checks all combinations.
for (i in my.nu){

  print(i)
  for (j in my.gamma){

    print(j)
    svm.model<-svm(Data.PCA.Prcomp[,1:100],y=NULL,
                type='one-classification',
                nu=i,
                scale=FALSE,
                kernel="radial",
                gamma = j)
    oneclass.d.values <-cbind(oneclass.d.values,svm.model$decision.values)
    column.name <- paste('nu =', i, ',gamma =',j, sep = " ")
```
```

```
    names(oneclass.d.values)[ncol(oneclass.d.values)] <- column.name
  }
}

write.csv(cbind(Data$X,oneclass.d.values), file =
  "onclassdvalsX3.csv",row.names=FALSE)

sort(svm.model$coefs)

```

Including Plots

```{r plot distributions, echo=FALSE}

search1 <- read.csv("onclassdvalsX.csv")
#my.nu    <- c(0.01, 0.05, 0.1, 0.5, 1)
#my.gamma <- c(0.01, 0.05, 0.1, 0.5, 1)

ggplot(search1, aes(search1$nu...0.1..gamma...0.01)) + geom_histogram() +
  labs(x =NULL , y = NULL)
ggsave("nu01gamma001.pdf")
ggplot(search1, aes(search1$nu...0.1..gamma...1)) + geom_histogram() +
  labs(x =NULL , y = NULL)
ggsave("nu01gamma1.pdf")
ggplot(search1, aes(search1$nu...0.5..gamma...0.05)) + geom_histogram() +
  labs(x =NULL , y = NULL)
ggsave("nu05gamma005.pdf")
ggplot(search1, aes(search1$nu...0.5..gamma...0.01)) + geom_histogram() +
  labs(x =NULL , y = NULL)
ggsave("nu05gamma001.pdf")


search2 <- read.csv("onclassdvalsX2.csv")
#my.nu    <- c(0.05,0.1,0.15)
#my.gamma <- c(0.001,0.005,0.01,0.02)

ggplot(search2, aes(search2$nu...0.005..gamma...0.001)) + geom_histogram() +
  labs(x =NULL , y = NULL)

ggplot(search2, aes(search2$nu...0.005..gamma...0.005)) + geom_histogram() +
  labs(x =NULL , y = NULL)

ggplot(search2, aes(search2$nu...0.005..gamma...0.01)) + geom_histogram() +
  labs(x =NULL , y = NULL)

ggplot(search2, aes(search2$nu...0.005..gamma...0.02)) + geom_histogram() +
  labs(x =NULL , y = NULL)
```

```
ggplot(search2, aes(search2$nu...0.1..gamma...0.001)) + geom_histogram() +
labs(x =NULL , y = NULL)
ggsave("nu01gamma0001.pdf")

ggplot(search2, aes(search2$nu...0.1..gamma...0.005)) + geom_histogram() +
  labs(x =NULL , y = NULL)
ggsave("nu01gamma0005.pdf")
ggplot(search2, aes(search2$nu...0.1..gamma...0.01)) + geom_histogram() +
  labs(x =NULL , y = NULL)

ggplot(search2, aes(search2$nu...0.1..gamma...0.02)) + geom_histogram() +
  labs(x =NULL , y = NULL)

ggplot(search2, aes(search2$nu...0.15..gamma...0.001)) + geom_histogram() +
  labs(x ="1" , y = NULL)
ggsave("nu015gamma0001.pdf")
ggplot(search2, aes(search2$nu...0.15..gamma...0.005)) + geom_histogram() +
  labs(x =NULL , y = NULL)
ggsave("nu015gamma0005.pdf")
ggplot(search2, aes(search2$nu...0.15..gamma...0.01)) + geom_histogram() +
  labs(x =NULL , y = NULL)

ggplot(search2, aes(search2$nu...0.15..gamma...0.02)) + geom_histogram() +
  labs(x =NULL , y = NULL)


search3 <- read.csv("onclassdvalsX3.csv")
#my.nu    <- c(0.13,0.18,0.2)
#my.gamma <- c(0.0007,0.002,0.0055)

ggplot(search3, aes(search3$nu...0.13..gamma...7e.04)) + geom_histogram() +
  labs(x =NULL , y = NULL)
#ggsave("nu013gamma0007.pdf")
ggplot(search3, aes(search3$nu...0.13..gamma...0.002)) + geom_histogram() +
  labs(x =NULL , y = NULL)
#ggsave("nu013gamma0002.pdf")
ggplot(search3, aes(search3$nu...0.13..gamma...0.0055)) + geom_histogram() +
  labs(x =NULL , y = NULL)

ggplot(search3, aes(search3$nu...0.18..gamma...7e.04)) + geom_histogram() +
  labs(x =NULL , y = NULL)
#ggsave("nu018gamma0007.pdf")
ggplot(search3, aes(search3$nu...0.18..gamma...0.002)) + geom_histogram() +
  labs(x =NULL , y = NULL)
#ggsave("nu018gamma0002.pdf")
ggplot(search3, aes(search3$nu...0.18..gamma...0.0055)) + geom_histogram() +
  labs(x =NULL , y = NULL)

ggplot(search3, aes(search3$nu...0.2..gamma...7e.04)) + geom_histogram() +
  labs(x =NULL , y = NULL)
```

```
ggplot(search3, aes(search3$nu...0.2..gamma...0.002)) + geom_histogram() +
  labs(x =NULL , y = NULL)

ggplot(search3, aes(search3$nu...0.2..gamma...0.0055)) + geom_histogram() +
  labs(x ="1" , y = NULL)


ggsave("nu015gamma0005.pdf")


```

````
```{r compare last two  top 100, echo=FALSE}
# compare "nu013gamma0002.pdf" and "nu018gamma0002.pdf""
ocsvmtop100_13_002 <- arrange(search3, search3$nu...0.13..gamma...0.002)
ocsvmtop100_18_002 <- arrange(search3, search3$nu...0.18..gamma...0.002)
sum(as.vector(ocsvmtop100_13_002 [1:100,1]) %in%
  as.vector(ocsvmtop100_18_002 [1:100,1]))

```


```{r compare to KNN top 100, echo=FALSE}
ocsvmtop100 <- arrange(search3, search3$nu...0.13..gamma...7e.04)
sum(as.vector(ocsvmtop100 [1:100,1]) %in% knntop100$OutlierWeightKNN50)
ocsvmtop100 <- arrange(search3, search3$nu...0.13..gamma...0.002)
sum(as.vector(ocsvmtop100 [1:100,1]) %in% knntop100$OutlierWeightKNN50)
ocsvmtop100 <- arrange(search3, search3$nu...0.13..gamma...0.0055)
sum(as.vector(ocsvmtop100 [1:100,1]) %in% knntop100$OutlierWeightKNN50)
ocsvmtop100 <- arrange(search3, search3$nu...0.18..gamma...7e.04)
sum(as.vector(ocsvmtop100 [1:100,1]) %in% knntop100$OutlierWeightKNN50)
ocsvmtop100 <- arrange(search3, search3$nu...0.18..gamma...0.002)
sum(as.vector(ocsvmtop100 [1:100,1]) %in% knntop100$OutlierWeightKNN50)
ocsvmtop100 <- arrange(search3, search3$nu...0.18..gamma...0.0055)
sum(as.vector(ocsvmtop100 [1:100,1]) %in% knntop100$OutlierWeightKNN50)
ocsvmtop100 <- arrange(search3, search3$nu...0.2..gamma...7e.04)
sum(as.vector(ocsvmtop100 [1:100,1]) %in% knntop100$OutlierWeightKNN50)
ocsvmtop100 <- arrange(search3, search3$nu...0.2..gamma...0.002)
sum(as.vector(ocsvmtop100 [1:100,1]) %in% knntop100$OutlierWeightKNN50)
ocsvmtop100 <- arrange(search3, search3$nu...0.2..gamma...0.0055)
sum(as.vector(ocsvmtop100 [1:100,1]) %in% knntop100$OutlierWeightKNN50)
```

```{r KNN as point anomalies based on Node grouping, echo = Falsesd}
# As discovered earlier, the anomalies need to be considered in context.
# To move from contextual anomaly detection to point anomaly detection,
# the data set will be broken up into subsets. Various combinations of
# subsets will be investigated.
````
````

```
Data.PCA100 <- cbind(Data[,c(1:9)],Data.PCA.Prcomp[,1:100])

# A nested for loop calculated the SVM decision values per Node subset
# and then adds the results to the original dataset
# First Search
my.nu    <- c(0.13,0.18,0.2)
my.gamma <- c(0.0007,0.002,0.0055)

# Second search
my.nu    <- c(0.3,0.4,0.5)
my.gamma <- c(0.0001,0.0002,0.0003)


Nodes <- unique(Data.PCA100$SGW_ID)
SVM.Decision.values <- Data.PCA100$X
SVM.Decision.values <- as.data.frame(SVM.Decision.values)
SVM.Decision.values <- SVM.Decision.values[-1]
#SVMData.index <- c(1:110)

for (i in my.nu){
  for (j in my.gamma){

    SVMData.index <- c(1:110)
    for (Node in Nodes){
      SVMsubset <- Data.PCA100 %>%
        filter(SGW_ID == Node) %>%
        select( c(10:109)) %>%
        svm(y=NULL,
            type='one-classification',
            nu=i,
            scale=FALSE,
            kernel="radial",
            gamma = j)

      SVMindex  <- Data.PCA100 %>%
      filter(SGW_ID == Node) %>%
      select(X)
      SVMData.index <- rbind(SVMData.index,
        cbind (SVMindex, SVMsubset$decision.values))
    }
    SVM.Decision.values <- cbind(SVM.Decision.values, SVMData.index[-1,] )
    column.name <- paste('nu =', i, ',gamma =',j, sep = " ")
    print(column.name)
    print(ncol(SVM.Decision.values))
    names(SVM.Decision.values)[ncol(SVM.Decision.values)] <- column.name
    }
}
# Remove duplicate X values
SVM.Decision.values <- SVM.Decision.values[,c(1,2,4,6,8,10,12,14,16,18)]
```

```
write.csv(SVM.Decision.values, file ="Node OCSVM Search 1.csv")

ggplot(SVM.Decision.values,
  aes(SVM.Decision.values$`nu = 0.3 ,gamma = 1e-04`)) +
    geom_histogram() + labs(x =NULL , y = NULL) + scale_y_sqrt()
ggsave("nu03gamma00001node.pdf")
ggplot(SVM.Decision.values,
  aes(SVM.Decision.values$`nu = 0.3 ,gamma = 2e-04`)) +
    geom_histogram() + labs(x =NULL , y = NULL) + scale_y_sqrt()
ggsave("nu03gamma00002node.pdf")
ggplot(SVM.Decision.values,
  aes(SVM.Decision.values$`nu = 0.3 ,gamma = 3e-04`)) +
    geom_histogram() + labs(x =NULL , y = NULL) + scale_y_sqrt()
ggsave("nu03gamma00003node.pdf")
ggplot(SVM.Decision.values,
  aes(SVM.Decision.values$`nu = 0.4 ,gamma = 1e-04`)) +
    geom_histogram() + labs(x =NULL , y = NULL) + scale_y_sqrt()
ggsave("nu04amma00001node.pdf")
ggplot(SVM.Decision.values,
  aes(SVM.Decision.values$`nu = 0.4 ,gamma = 2e-04`)) +
    geom_histogram() + labs(x =NULL , y = NULL) + scale_y_sqrt()
ggsave("nu04gamma00002node.pdf")
ggplot(SVM.Decision.values,
  aes(SVM.Decision.values$`nu = 0.4 ,gamma = 3e-04`)) +
    geom_histogram() + labs(x =NULL , y = NULL) + scale_y_sqrt()
ggsave("nu04gamma00003node.pdf")
ggplot(SVM.Decision.values,
  aes(SVM.Decision.values$`nu = 0.5 ,gamma = 1e-04`)) +
    geom_histogram() + labs(x =NULL , y = NULL) + scale_y_sqrt()
ggsave("nu05gamma00001node.pdf")
ggplot(SVM.Decision.values,
  aes(SVM.Decision.values$`nu = 0.5 ,gamma = 2e-04`)) +
    geom_histogram() + labs(x =NULL , y = NULL) + scale_y_sqrt()
ggsave("nu05gamma00002node.pdf")
ggplot(SVM.Decision.values,
  aes(SVM.Decision.values$`nu = 0.5 ,gamma = 3e-04`)) +
    geom_histogram() + labs(x =NULL , y = NULL) + scale_y_sqrt()
ggsave("nu05gamma00003node.pdf")


SVMnode.vs.all <- matrix(rep(1:27),ncol =3)
top100SVMnode <- arrange(SVM.Decision.values,
  SVM.Decision.values$`nu = 0.3 ,gamma = 1e-04`)
SVMnode.vs.all[1,1] <-sum(as.vector(top100SVMnode[1:100,1]) %in%
  as.vector(ocsvmtop100_13_002[1:100,1]))
SVMnode.vs.all[1,2] <-sum(as.vector(top100SVMnode[1:200,1]) %in%
  as.vector(ocsvmtop100_13_002[1:200,1]))
SVMnode.vs.all[1,3] <-sum(as.vector(top100SVMnode[1:300,1]) %in%
  as.vector(ocsvmtop100_13_002[1:300,1]))
top100SVMnode <- arrange(SVM.Decision.values,
```

```
  SVM.Decision.values$`nu = 0.3 ,gamma = 2e-04`)
SVMnode.vs.all[2,1] <- sum(as.vector(top100SVMnode[1:100,1]) %in%
  as.vector(ocsvmtop100_13_002[1:100,1]))
SVMnode.vs.all[2,2] <- sum(as.vector(top100SVMnode[1:200,1]) %in%
  as.vector(ocsvmtop100_13_002[1:200,1]))
SVMnode.vs.all[2,3] <- sum(as.vector(top100SVMnode[1:300,1]) %in%
  as.vector(ocsvmtop100_13_002[1:300,1]))
top100SVMnode <- arrange(SVM.Decision.values,
  SVM.Decision.values$`nu = 0.3 ,gamma = 3e-04`)
SVMnode.vs.all[3,1] <- sum(as.vector(top100SVMnode[1:100,1]) %in%
  as.vector(ocsvmtop100_13_002[1:100,1]))
SVMnode.vs.all[3,2] <- sum(as.vector(top100SVMnode[1:200,1]) %in%
  as.vector(ocsvmtop100_13_002[1:200,1]))
SVMnode.vs.all[3,3] <- sum(as.vector(top100SVMnode[1:300,1]) %in%
  as.vector(ocsvmtop100_13_002[1:300,1]))
top100SVMnode <- arrange(SVM.Decision.values,
  SVM.Decision.values$`nu = 0.4 ,gamma = 1e-04`)
SVMnode.vs.all[4,1] <- sum(as.vector(top100SVMnode[1:100,1]) %in%
  as.vector(ocsvmtop100_13_002[1:100,1]))
SVMnode.vs.all[4,2] <- sum(as.vector(top100SVMnode[1:200,1]) %in%
  as.vector(ocsvmtop100_13_002[1:200,1]))
SVMnode.vs.all[4,3] <- sum(as.vector(top100SVMnode[1:300,1]) %in%
  as.vector(ocsvmtop100_13_002[1:300,1]))
top100SVMnode <- arrange(SVM.Decision.values,
  SVM.Decision.values$`nu = 0.4 ,gamma = 2e-04`)
SVMnode.vs.all[5,1] <- sum(as.vector(top100SVMnode[1:100,1]) %in%
  as.vector(ocsvmtop100_13_002[1:100,1]))
SVMnode.vs.all[5,2] <- sum(as.vector(top100SVMnode[1:200,1]) %in%
  as.vector(ocsvmtop100_13_002[1:200,1]))
SVMnode.vs.all[5,3] <- sum(as.vector(top100SVMnode[1:300,1]) %in%
  as.vector(ocsvmtop100_13_002[1:300,1]))
top100SVMnode <- arrange(SVM.Decision.values,
  SVM.Decision.values$`nu = 0.4 ,gamma = 3e-04`)
SVMnode.vs.all[6,1] <- sum(as.vector(top100SVMnode[1:100,1]) %in%
  as.vector(ocsvmtop100_13_002[1:100,1]))
SVMnode.vs.all[6,2] <- sum(as.vector(top100SVMnode[1:200,1]) %in%
  as.vector(ocsvmtop100_13_002[1:200,1]))
SVMnode.vs.all[6,3] <- sum(as.vector(top100SVMnode[1:300,1]) %in%
  as.vector(ocsvmtop100_13_002[1:300,1]))
top100SVMnode <- arrange(SVM.Decision.values,
  SVM.Decision.values$`nu = 0.5 ,gamma = 1e-04`)
SVMnode.vs.all[7,1] <- sum(as.vector(top100SVMnode[1:100,1]) %in%
  as.vector(ocsvmtop100_13_002[1:100,1]))
SVMnode.vs.all[7,2] <- sum(as.vector(top100SVMnode[1:200,1]) %in%
  as.vector(ocsvmtop100_13_002[1:200,1]))
SVMnode.vs.all[7,3] <- sum(as.vector(top100SVMnode[1:300,1]) %in%
  as.vector(ocsvmtop100_13_002[1:300,1]))
top100SVMnode <- arrange(SVM.Decision.values,
  SVM.Decision.values$`nu = 0.5 ,gamma = 2e-04`)
SVMnode.vs.all[8,1] <- sum(as.vector(top100SVMnode[1:100,1]) %in%
```

```
  as.vector(ocsvmtop100_13_002[1:100,1]))
SVMnode.vs.all[8,2] <- sum(as.vector(top100SVMnode[1:200,1]) %in%
  as.vector(ocsvmtop100_13_002[1:200,1]))
SVMnode.vs.all[8,3] <- sum(as.vector(top100SVMnode[1:300,1]) %in%
  as.vector(ocsvmtop100_13_002[1:300,1]))
top100SVMnode <- arrange(SVM.Decision.values,
  SVM.Decision.values$`nu = 0.5 ,gamma = 3e-04`)
SVMnode.vs.all[9,1] <- sum(as.vector(top100SVMnode[1:100,1]) %in%
  as.vector(ocsvmtop100_13_002[1:100,1]))
SVMnode.vs.all[9,2] <- sum(as.vector(top100SVMnode[1:200,1]) %in%
  as.vector(ocsvmtop100_13_002[1:200,1]))
SVMnode.vs.all[9,3] <- sum(as.vector(top100SVMnode[1:300,1]) %in%
  as.vector(ocsvmtop100_13_002[1:300,1]))
```

```
```{r KNN as point anomalies based on Node grouping, echo = Falsesd}
# As discovered earlier, the anomalies need to be considered in context.
# To move from contextual anomaly detection to point anomaly detection,
# the data set will be broken up into subsets. Various combinations of
# subsets will be investigated.

Data.PCA100 <- cbind(Data[,c(1:9)],Data.PCA.Prcomp[,1:100])


# A nested for loop calculated the SVM decision values per Node subset
# and then adds the results to the original dataset
# First Search

my.nu    <- c(0.3,0.4,0.5)
my.gamma <- c(0.0001,0.0002,0.0003)


Nodes <- unique(Data.PCA100$SGW_ID)
Times <- unique(Data.PCA100$Time)

SVM.Decision.values <- Data.PCA100$X
SVM.Decision.values <- as.data.frame(SVM.Decision.values)
SVM.Decision.values <- SVM.Decision.values[-1]
#SVMData.index <- c(1:110)

for (i in my.nu){
  for (j in my.gamma){

    SVMData.index <- c(1:110)
    for (Node in Nodes){
      for (TOD in Times){
      SVMsubset <- Data.PCA100 %>%
        filter(SGW_ID == Node & Time ==TOD) %>%
        select( c(10:109)) %>%
        svm(y=NULL,
```
```

```
            type='one-classification',
            nu=i,
            scale=FALSE,
            kernel="radial",
            gamma = j)

      SVMindex  <- Data.PCA100 %>%
      filter(SGW_ID == Node& Time == TOD) %>%
      select(X)
      SVMData.index <- rbind(SVMData.index,
        cbind (SVMindex, SVMsubset$decision.values))
      }
    }
    SVM.Decision.values <- cbind(SVM.Decision.values, SVMData.index[-1,] )
    column.name <- paste('nu =', i, ',gamma =',j, sep = " ")
    names(SVM.Decision.values)[ncol(SVM.Decision.values)] <- column.name
    }
}
# Remove duplicate X values
SVM.Decision.values <- SVM.Decision.values[,c(1,2,4,6,8,10,12,14,16,18)]

write.csv(SVM.Decision.values, file = "Node Time OCSVM Search 1.csv")

SVMnode.vs.all <- matrix(rep(1:27),ncol =3)
top100SVMnode <- arrange(SVM.Decision.values,
 SVM.Decision.values$`nu = 0.3 ,gamma = 1e-04`)
SVMnode.vs.all[1,1] <-sum(as.vector(top100SVMnode[1:100,1]) %in%
  as.vector(ocsvmtop100_13_002[1:100,1]))
SVMnode.vs.all[1,2] <-sum(as.vector(top100SVMnode[1:200,1]) %in%
  as.vector(ocsvmtop100_13_002[1:200,1]))
SVMnode.vs.all[1,3] <-sum(as.vector(top100SVMnode[1:300,1]) %in%
  as.vector(ocsvmtop100_13_002[1:300,1]))
top100SVMnode <- arrange(SVM.Decision.values,
  SVM.Decision.values$`nu = 0.3 ,gamma = 2e-04`)
SVMnode.vs.all[2,1] <- sum(as.vector(top100SVMnode[1:100,1]) %in%
  as.vector(ocsvmtop100_13_002[1:100,1]))
SVMnode.vs.all[2,2] <- sum(as.vector(top100SVMnode[1:200,1]) %in%
  as.vector(ocsvmtop100_13_002[1:200,1]))
SVMnode.vs.all[2,3] <- sum(as.vector(top100SVMnode[1:300,1]) %in%
  as.vector(ocsvmtop100_13_002[1:300,1]))
top100SVMnode <- arrange(SVM.Decision.values,
  SVM.Decision.values$`nu = 0.3 ,gamma = 3e-04`)
SVMnode.vs.all[3,1] <- sum(as.vector(top100SVMnode[1:100,1]) %in%
  as.vector(ocsvmtop100_13_002[1:100,1]))
SVMnode.vs.all[3,2] <- sum(as.vector(top100SVMnode[1:200,1]) %in%
  as.vector(ocsvmtop100_13_002[1:200,1]))
SVMnode.vs.all[3,3] <- sum(as.vector(top100SVMnode[1:300,1]) %in%
  as.vector(ocsvmtop100_13_002[1:300,1]))
top100SVMnode <- arrange(SVM.Decision.values,
  SVM.Decision.values$`nu = 0.4 ,gamma = 1e-04`)
```

```
SVMnode.vs.all[4,1] <- sum(as.vector(top100SVMnode[1:100,1]) %in%
  as.vector(ocsvmtop100_13_002[1:100,1]))
SVMnode.vs.all[4,2] <- sum(as.vector(top100SVMnode[1:200,1]) %in%
  as.vector(ocsvmtop100_13_002[1:200,1]))
SVMnode.vs.all[4,3] <- sum(as.vector(top100SVMnode[1:300,1]) %in%
  as.vector(ocsvmtop100_13_002[1:300,1]))
top100SVMnode <- arrange(SVM.Decision.values,
  SVM.Decision.values$`nu = 0.4 ,gamma = 2e-04`)
SVMnode.vs.all[5,1] <- sum(as.vector(top100SVMnode[1:100,1]) %in%
  as.vector(ocsvmtop100_13_002[1:100,1]))
SVMnode.vs.all[5,2] <- sum(as.vector(top100SVMnode[1:200,1]) %in%
  as.vector(ocsvmtop100_13_002[1:200,1]))
SVMnode.vs.all[5,3] <- sum(as.vector(top100SVMnode[1:300,1]) %in%
  as.vector(ocsvmtop100_13_002[1:300,1]))
top100SVMnode <- arrange(SVM.Decision.values,
  SVM.Decision.values$`nu = 0.4 ,gamma = 3e-04`)
SVMnode.vs.all[6,1] <- sum(as.vector(top100SVMnode[1:100,1]) %in%
  as.vector(ocsvmtop100_13_002[1:100,1]))
SVMnode.vs.all[6,2] <- sum(as.vector(top100SVMnode[1:200,1]) %in%
  as.vector(ocsvmtop100_13_002[1:200,1]))
SVMnode.vs.all[6,3] <- sum(as.vector(top100SVMnode[1:300,1]) %in%
  as.vector(ocsvmtop100_13_002[1:300,1]))
top100SVMnode <- arrange(SVM.Decision.values,
  SVM.Decision.values$`nu = 0.5 ,gamma = 1e-04`)
SVMnode.vs.all[7,1] <- sum(as.vector(top100SVMnode[1:100,1]) %in%
  as.vector(ocsvmtop100_13_002[1:100,1]))
SVMnode.vs.all[7,2] <- sum(as.vector(top100SVMnode[1:200,1]) %in%
  as.vector(ocsvmtop100_13_002[1:200,1]))
SVMnode.vs.all[7,3] <- sum(as.vector(top100SVMnode[1:300,1]) %in%
  as.vector(ocsvmtop100_13_002[1:300,1]))
top100SVMnode <- arrange(SVM.Decision.values,
  SVM.Decision.values$`nu = 0.5 ,gamma = 2e-04`)
SVMnode.vs.all[8,1] <- sum(as.vector(top100SVMnode[1:100,1]) %in%
  as.vector(ocsvmtop100_13_002[1:100,1]))
SVMnode.vs.all[8,2] <- sum(as.vector(top100SVMnode[1:200,1]) %in%
  as.vector(ocsvmtop100_13_002[1:200,1]))
SVMnode.vs.all[8,3] <- sum(as.vector(top100SVMnode[1:300,1]) %in%
  as.vector(ocsvmtop100_13_002[1:300,1]))
top100SVMnode <- arrange(SVM.Decision.values,
  SVM.Decision.values$`nu = 0.5 ,gamma = 3e-04`)
SVMnode.vs.all[9,1] <- sum(as.vector(top100SVMnode[1:100,1]) %in%
  as.vector(ocsvmtop100_13_002[1:100,1]))
SVMnode.vs.all[9,2] <- sum(as.vector(top100SVMnode[1:200,1]) %in%
  as.vector(ocsvmtop100_13_002[1:200,1]))
SVMnode.vs.all[9,3] <- sum(as.vector(top100SVMnode[1:300,1]) %in%
  as.vector(ocsvmtop100_13_002[1:300,1]))

ggplot(SVM.Decision.values,
  aes(SVM.Decision.values$`nu = 0.3 ,gamma = 1e-04`)) +
    geom_histogram() + labs(x =NULL , y = NULL) + scale_y_sqrt()
```

```
ggsave("nu03gamma00001nodetime.pdf")
ggplot(SVM.Decision.values,
  aes(SVM.Decision.values$`nu = 0.3 ,gamma = 2e-04`)) +
    geom_histogram() + labs(x =NULL , y = NULL) + scale_y_sqrt()
ggsave("nu03gamma00002nodetime.pdf")
ggplot(SVM.Decision.values,
  aes(SVM.Decision.values$`nu = 0.3 ,gamma = 3e-04`)) + geom_histogram() +
  labs(x =NULL , y = NULL) + scale_y_sqrt()
ggsave("nu03gamma00003nodetime.pdf")
ggplot(SVM.Decision.values,
  aes(SVM.Decision.values$`nu = 0.4 ,gamma = 1e-04`)) +
  geom_histogram() + labs(x =NULL , y = NULL) + scale_y_sqrt()
ggsave("nu04amma00001nodetime.pdf")
ggplot(SVM.Decision.values,
  aes(SVM.Decision.values$`nu = 0.4 ,gamma = 2e-04`)) +
  geom_histogram() + labs(x =NULL , y = NULL) + scale_y_sqrt()
ggsave("nu04gamma00002nodetime.pdf")
ggplot(SVM.Decision.values,
aes(SVM.Decision.values$`nu = 0.4 ,gamma = 3e-04`)) + geom_histogram() +
  labs(x =NULL , y = NULL) + scale_y_sqrt()
ggsave("nu04gamma00003nodetime.pdf")
ggplot(SVM.Decision.values,
aes(SVM.Decision.values$`nu = 0.5 ,gamma = 1e-04`)) + geom_histogram() +
labs(x =NULL , y = NULL) + scale_y_sqrt()
ggsave("nu05gamma00001nodetime.pdf")
ggplot(SVM.Decision.values,
aes(SVM.Decision.values$`nu = 0.5 ,gamma = 2e-04`)) + geom_histogram() +
labs(x =NULL , y = NULL) + scale_y_sqrt()
ggsave("nu05gamma00002nodetime.pdf")
ggplot(SVM.Decision.values,
aes(SVM.Decision.values$`nu = 0.5 ,gamma = 3e-04`)) + geom_histogram() +
labs(x =NULL , y = NULL) + scale_y_sqrt()
ggsave("nu05gamma00003nodetime.pdf")

node.ocsvm <- read.csv("Node OCSVM Search 2.csv", header = TRUE, sep = ",")
node.ocsvm <- arrange(node.ocsvm, node.ocsvm$nu...0.3..gamma...3e.04)

SVMnode.vs.all <- matrix(rep(1:27),ncol =3)
top100SVMnode <- arrange(SVM.Decision.values,
  SVM.Decision.values$`nu = 0.3 ,gamma = 1e-04`)
SVMnode.vs.all[1,1] <-sum(as.vector(top100SVMnode[1:100,1]) %in%
  as.vector(node.ocsvm[1:100,2]))
SVMnode.vs.all[1,2] <-sum(as.vector(top100SVMnode[1:200,1]) %in%
  as.vector(node.ocsvm[1:200,2]))
SVMnode.vs.all[1,3] <-sum(as.vector(top100SVMnode[1:300,1]) %in%
  as.vector(node.ocsvm[1:300,2]))
top100SVMnode <- arrange(SVM.Decision.values,
  SVM.Decision.values$`nu = 0.3 ,gamma = 2e-04`)
SVMnode.vs.all[2,1] <- sum(as.vector(top100SVMnode[1:100,1]) %in%
  as.vector(node.ocsvm[1:100,2]))
```

```
SVMnode.vs.all[2,2] <- sum(as.vector(top100SVMnode[1:200,1]) %in%
  as.vector(node.ocsvm[1:200,2]))
SVMnode.vs.all[2,3] <- sum(as.vector(top100SVMnode[1:300,1]) %in%
  as.vector(node.ocsvm[1:300,2]))
top100SVMnode <- arrange(SVM.Decision.values,
  SVM.Decision.values$`nu = 0.3 ,gamma = 3e-04`)
SVMnode.vs.all[3,1] <- sum(as.vector(top100SVMnode[1:100,1]) %in%
  as.vector(node.ocsvm[1:100,2]))
SVMnode.vs.all[3,2] <- sum(as.vector(top100SVMnode[1:200,1]) %in%
  as.vector(node.ocsvm[1:200,2]))
SVMnode.vs.all[3,3] <- sum(as.vector(top100SVMnode[1:300,1]) %in%
  as.vector(node.ocsvm[1:300,2]))
top100SVMnode <- arrange(SVM.Decision.values,
  SVM.Decision.values$`nu = 0.4 ,gamma = 1e-04`)
SVMnode.vs.all[4,1] <- sum(as.vector(top100SVMnode[1:100,1]) %in%
  as.vector(node.ocsvm[1:100,2]))
SVMnode.vs.all[4,2] <- sum(as.vector(top100SVMnode[1:200,1]) %in%
  as.vector(node.ocsvm[1:200,2]))
SVMnode.vs.all[4,3] <- sum(as.vector(top100SVMnode[1:300,1]) %in%
  as.vector(node.ocsvm[1:300,2]))
top100SVMnode <- arrange(SVM.Decision.values,
  SVM.Decision.values$`nu = 0.4 ,gamma = 2e-04`)
SVMnode.vs.all[5,1] <- sum(as.vector(top100SVMnode[1:100,1]) %in%
  as.vector(node.ocsvm[1:100,2]))
SVMnode.vs.all[5,2] <- sum(as.vector(top100SVMnode[1:200,1]) %in%
  as.vector(node.ocsvm[1:200,2]))
SVMnode.vs.all[5,3] <- sum(as.vector(top100SVMnode[1:300,1]) %in%
  as.vector(node.ocsvm[1:300,2]))
top100SVMnode <- arrange(SVM.Decision.values,
  SVM.Decision.values$`nu = 0.4 ,gamma = 3e-04`)
SVMnode.vs.all[6,1] <- sum(as.vector(top100SVMnode[1:100,1]) %in%
  as.vector(node.ocsvm[1:100,2]))
SVMnode.vs.all[6,2] <- sum(as.vector(top100SVMnode[1:200,1]) %in%
  as.vector(node.ocsvm[1:200,2]))
SVMnode.vs.all[6,3] <- sum(as.vector(top100SVMnode[1:300,1]) %in%
  as.vector(node.ocsvm[1:300,2]))
top100SVMnode <- arrange(SVM.Decision.values,
  SVM.Decision.values$`nu = 0.5 ,gamma = 1e-04`)
SVMnode.vs.all[7,1] <- sum(as.vector(top100SVMnode[1:100,1]) %in%
  as.vector(node.ocsvm[1:100,2]))
SVMnode.vs.all[7,2] <- sum(as.vector(top100SVMnode[1:200,1]) %in%
  as.vector(node.ocsvm[1:200,2]))
SVMnode.vs.all[7,3] <- sum(as.vector(top100SVMnode[1:300,1]) %in%
  as.vector(node.ocsvm[1:300,2]))
top100SVMnode <- arrange(SVM.Decision.values,
  SVM.Decision.values$`nu = 0.5 ,gamma = 2e-04`)
SVMnode.vs.all[8,1] <- sum(as.vector(top100SVMnode[1:100,1]) %in%
  as.vector(node.ocsvm[1:100,2]))
SVMnode.vs.all[8,2] <- sum(as.vector(top100SVMnode[1:200,1]) %in%
  as.vector(node.ocsvm[1:200,2]))
```

```
SVMnode.vs.all[8,3] <- sum(as.vector(top100SVMnode[1:300,1]) %in%
  as.vector(node.ocsvm[1:300,2]))
top100SVMnode <- arrange(SVM.Decision.values,
  SVM.Decision.values$‘nu = 0.5 ,gamma = 3e-04‘)
SVMnode.vs.all[9,1] <- sum(as.vector(top100SVMnode[1:100,1]) %in%
  as.vector(node.ocsvm[1:100,2]))
SVMnode.vs.all[9,2] <- sum(as.vector(top100SVMnode[1:200,1]) %in%
  as.vector(node.ocsvm[1:200,2]))
SVMnode.vs.all[9,3] <- sum(as.vector(top100SVMnode[1:300,1]) %in%
  as.vector(node.ocsvm[1:300,2]))
‘‘‘


‘‘‘{r Comparing KNN approaches to OCSVM appoaches, echo = Falsesd}

# Here we compare the 3 KNN approaches with the 3 OCSVM approaches.
# For OCSVM non-subset approach the $\nu$ set to  0.13 and $\gamma$ s
# et to 0.002 results are used.
# for the OCSVM node and node-time based results, the $\nu$ to 0.5 and
# $\gamma$ to 0.0001 results are used.
# For KNN the Weighted 50KNN results are used for non-subset and node based.
# For the KNN node-time approach the weight of 8 was used.

node.time.ocsvm <-
  read.csv("NodeTimeOCSVMSearch1.csv", header = TRUE, sep = ",")
node.time.ocsvm <-
  arrange(node.time.ocsvm, node.time.ocsvm$nu...0.5..gamma...1e.04)

node.ocsvm <- read.csv("Node OCSVM Search 2.csv", header = TRUE, sep = ",")
node.ocsvm <- arrange(node.ocsvm, node.ocsvm$nu...0.5..gamma...1e.04)

ocsvm <- read.csv("onclassdvalsX3.csv", header = TRUE, sep = ",")
ocsvm <- arrange(ocsvm, ocsvm$nu...0.13..gamma...0.002)

node.time.knn <- read.csv("NodeTimeKNN.csv", header = TRUE, sep = ",")
node.time.knn <- arrange(node.time.knn,
  desc(node.time.knn$NodeTimeWeight8))

node.knn <- read.csv("NodeKNN.csv", header = TRUE, sep = ",")
node.knn <- arrange(node.knn, desc(node.knn$NodeWeight50))

my.knn <- read.csv("KNN.csv", header = TRUE, sep = "," )
my.knn <- arrange(my.knn, desc(my.knn$Weight50))

sum(as.vector(my.knn[1:100,2]) %in% as.vector(ocsvm[1:100,1]))
sum(as.vector(my.knn[1:100,2]) %in% as.vector(node.ocsvm [1:100,2]))
sum(as.vector(my.knn[1:100,2]) %in%
  as.vector(node.time.ocsvm [1:100,2]))

sum(as.vector(node.knn[1:100,2]) %in% as.vector(ocsvm[1:100,1]))
```

```
sum(as.vector(node.knn[1:100,2]) %in%
  as.vector(node.ocsvm [1:100,2]))
sum(as.vector(node.knn[1:100,2]) %in%
  as.vector(node.time.ocsvm [1:100,2]))

sum(as.vector(node.time.knn[1:100,2]) %in% as.vector(ocsvm[1:100,1]))
sum(as.vector(node.time.knn[1:100,2]) %in%
  as.vector(node.ocsvm [1:100,2]))
sum(as.vector(node.time.knn[1:100,2]) %in%
  as.vector(node.time.ocsvm [1:100,2]))
```

B.5 Local Outlier Factor

```
---
title: "SGW LOF"
author: "Jason Salzwedel"
---

#Describe LOF
```{r local oulier, include=TRUE}
# Create two clusters of data one sparse and one dense
# Then Add two outliers, one local and one global
# Plot the results.

c1a <-runif(100,min=0, max =30)
c1b <-runif(100,min=0, max =30)

c1 <- cbind (c1a,c1b)


c2a <-rnorm(150,mean = 0, sd = 1)
c2b <-rnorm(150,mean = 0, sd = 1)
c2 <- cbind (c2a+115,c2b+15)
c3 <- rbind(c1,c2)
o1 <- 108
o2 <- 15
o <- cbind(o1,o2)

x1 <- 60
x2 <- 10
x <- cbind(x1,x2)

pdf("LocalOutlier.pdf",width=12,height=8)


plot(c3, pch = 20, xlab = NA, ylab = NA, axes = FALSE)
#box()
points(o)
```
```

```
points(x, pch = 4)

dev.off()
```


```{r Load data, include=TRUE}
library(dbscan)
library(ggplot2)
library(tidyverse)

Data <- read.csv("FinalDataSGW.CSV", header = TRUE, sep = ",")
Data.PCA.Prcomp <- read.csv("Data.PCA.Prcomp.csv", header = TRUE, sep = ",")
Data.PCA100 <- cbind(Data[,c(1:9)],Data.PCA.Prcomp[,1:100])
```

```{r Generate LOF values and Plot data, include=TRUE}

# Generate the LOF values based on the first 100 principle components
# for k values from 10 to 60.
LOFresults <- c(1:16070)

for (i in c(1:50)){
  loftemp <- lof(as.matrix(Data.PCA.Prcomp[,1:100]), k =i*5)
  LOFresults <-  cbind(LOFresults,loftemp)
}


# Rename the columns to show the K values used
LOFresults <- as.data.frame(LOFresults)
names(LOFresults)[2:51] <- c("LOF_k5", "LOF_k10", "LOF_k15",
  "LOF_k20", "LOF_k25", "LOF_k30",  LOF_k35", "LOF_k40", "LOF_k45",
  "LOF_k50", "LOF_k55", "LOF_k60", "LOF_k65", "LOF_k70", "LOF_k75",
  "LOF_k80", "LOF_k85", "LOF_k90", "LOF_k95", "LOF_k100", "LOF_k105",
  "LOF_k110", "LOF_k115", "LOF_k120", "LOF_k125", "LOF_k130",
  "LOF_k135", "LOF_k140", "LOF_k145", "LOF_k150", "LOF_k155",
  "LOF_k160", "LOF_k165", "LOF_k170", "LOF_k175", "LOF_k180",
  "LOF_k185", "LOF_k190", "LOF_k195", "LOF_k200", "LOF_k205",
  "LOF_k210", "LOF_k215", "LOF_k220", "LOF_k225", "LOF_k230",
  "LOF_k235", "LOF_k240", "LOF_k245", "LOF_k250"
                              )

write.csv(LOFresults, file = "LOFresults.csv",row.names=FALSE)

OFPlot <- read.csv("LOFresults.csv")

# Plot the results for investigation
ggplot(LOFPlot, aes(LOFPlot[,2])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk5.pdf")
```

```
ggplot(LOFPlot, aes(LOFPlot[,3])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk10.pdf")
ggplot(LOFPlot, aes(LOFPlot[,4])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk15.pdf")
ggplot(LOFPlot, aes(LOFPlot[,5])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk20.pdf")
ggplot(LOFPlot, aes(LOFPlot[,6])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk25.pdf")
ggplot(LOFPlot, aes(LOFPlot[,7])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk30.pdf")
ggplot(LOFPlot, aes(LOFPlot[,8])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk35.pdf")
ggplot(LOFPlot, aes(LOFPlot[,9])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk40.pdf")
ggplot(LOFPlot, aes(LOFPlot[,10])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk45.pdf")
ggplot(LOFPlot, aes(LOFPlot[,11])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk50.pdf")
ggplot(LOFPlot, aes(LOFPlot[,12])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk55.pdf")
ggplot(LOFPlot, aes(LOFPlot[,13])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk60.pdf")
ggplot(LOFPlot, aes(LOFPlot[,14])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk65.pdf")
ggplot(LOFPlot, aes(LOFPlot[,15])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk70.pdf")
ggplot(LOFPlot, aes(LOFPlot[,16])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk75.pdf")
ggplot(LOFPlot, aes(LOFPlot[,17])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk80.pdf")
ggplot(LOFPlot, aes(LOFPlot[,18])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk85.pdf")
ggplot(LOFPlot, aes(LOFPlot[,19])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
```

```
ggsave("LOFk90.pdf")
ggplot(LOFPlot, aes(LOFPlot[,20])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk95.pdf")
ggplot(LOFPlot, aes(LOFPlot[,21])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk100.pdf")
ggplot(LOFPlot, aes(LOFPlot[,22])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk105.pdf")
ggplot(LOFPlot, aes(LOFPlot[,23])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk110.pdf")
ggplot(LOFPlot, aes(LOFPlot[,24])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk115.pdf")
ggplot(LOFPlot, aes(LOFPlot[,25])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk120.pdf")
ggplot(LOFPlot, aes(LOFPlot[,26])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk125.pdf")
ggplot(LOFPlot, aes(LOFPlot[,27])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk130.pdf")
ggplot(LOFPlot, aes(LOFPlot[,28])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk135.pdf")
ggplot(LOFPlot, aes(LOFPlot[,29])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk140.pdf")
ggplot(LOFPlot, aes(LOFPlot[,30])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk145.pdf")
ggplot(LOFPlot, aes(LOFPlot[,31])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk150.pdf")
ggplot(LOFPlot, aes(LOFPlot[,32])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk155.pdf")
ggplot(LOFPlot, aes(LOFPlot[,33])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk160.pdf")
ggplot(LOFPlot, aes(LOFPlot[,34])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk165.pdf")
ggplot(LOFPlot, aes(LOFPlot[,35])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk170.pdf")
ggplot(LOFPlot, aes(LOFPlot[,36])) + geom_histogram() +
```

```
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk175.pdf")
ggplot(LOFPlot, aes(LOFPlot[,37])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk180.pdf")
ggplot(LOFPlot, aes(LOFPlot[,38])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk185.pdf")
ggplot(LOFPlot, aes(LOFPlot[,39])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk190.pdf")
ggplot(LOFPlot, aes(LOFPlot[,40])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk195.pdf")
ggplot(LOFPlot, aes(LOFPlot[,41])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk200.pdf")
ggplot(LOFPlot, aes(LOFPlot[,42])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk205.pdf")
ggplot(LOFPlot, aes(LOFPlot[,43])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk210.pdf")
ggplot(LOFPlot, aes(LOFPlot[,44])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk215.pdf")
ggplot(LOFPlot, aes(LOFPlot[,45])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk220.pdf")
ggplot(LOFPlot, aes(LOFPlot[,46])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk225.pdf")
ggplot(LOFPlot, aes(LOFPlot[,47])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk230.pdf")
ggplot(LOFPlot, aes(LOFPlot[,48])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk235.pdf")
ggplot(LOFPlot, aes(LOFPlot[,49])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk240.pdf")
ggplot(LOFPlot, aes(LOFPlot[,50])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk245.pdf")
ggplot(LOFPlot, aes(LOFPlot[,51])) + geom_histogram() +
  labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFk250.pdf")

write.csv(LOFPlot$LOF_k100, file = "LOF k100")
```

```{r compare results of different k values, include=TRUE}

# Create list of 100 results per k value
Top100Lof <- rep(0,100)
Top100Lof <- as.data.frame(Top100Lof)
for( i in 2:51) {
  temp <- arrange(LOFPlot, desc(LOFPlot[,i]))
  Top100Lof <- cbind(Top100Lof, temp[1:100,1])
    column.name <- paste('Lof_k=', i*5-5)
    print(column.name)
    names(Top100Lof)[ncol(Top100Lof)] <- column.name
}

Top100Lof <- Top100Lof[,2:51]

# Create a matrix comparing each of the differnt LOF approaches
# (each k value) with each other (how many match)

MatchMatrix <- matrix(rep(0,2500), ncol = 50) # template to hold results

i = 1
j = 1
for(i in 1:50){
  for(j in 1:50){
    matchmatch <- sum(Top100Lof[,i] %in% Top100Lof[,j])
    MatchMatrix[i,j] <- matchmatch
  }
}

MatchMatrix <- as.data.frame(MatchMatrix)

write.csv(MatchMatrix, file = "LOFcomp 5.csv")
```


```{r Generate LOF values and Plot data, include=TRUE}

# Generate the LOF values based on the first 100 principle
# components grouped by Node for k values from 10 to 60.

LOF.Node.results <- Data.PCA100$X
LOF.Node.results <- as.data.frame(LOF.Node.results)
LOF.Node.results <- LOF.Node.results[-1]

LOFresults <- c(1:16070)
Nodes <- unique(Data.PCA100$SGW_ID)

# Create a nested loop to run the LOF function on all
# subsets of data
```

```
for (i in c(1:50)){
    LOFData.index <- c(1:110)
    for (Node in Nodes){
      LOFsubset <- Data.PCA100 %>%
      filter(SGW_ID == Node) %>%
      select( c(10:109)) %>%
      lof(k =i*5)

      LOFindex  <- Data.PCA100 %>%
      filter(SGW_ID == Node) %>%
      select(X)
      LOFData.index <- rbind(LOFData.index, cbind (LOFindex, LOFsubset))

    }
    LOF.Node.results <- cbind(LOF.Node.results, LOFData.index[-1,] )
    column.name <- paste('k =', i*5)
    names(LOF.Node.results)[ncol(LOF.Node.results)] <- column.name
}

# Remove duplicate observation value columns

LOF.Node.results <- LOF.Node.results[,c(1, 2, 4, 6, 8,
  10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36,
  38, 40, 42, 44, 46, 48, 50, 52, 54, 56, 58, 60, 62, 64,
  66, 68, 70, 72, 74, 76, 78, 80, 82, 84, 86, 88, 90, 92,
  94, 96, 98, 100)]

write.csv(LOF.Node.results$`k = 100`, file ="Node LOF k100.csv")
```

```{r Compare Node based results of different k values, include=TRUE}

# Create list of 100 results per k value
Top100LofNode <- rep(0,100)
Top100LofNode <- as.data.frame(Top100LofNode)
for( i in 2:51) {
  temp <- arrange(LOF.Node.results, desc(LOF.Node.results[,i]))
  Top100LofNode <- cbind(Top100LofNode, temp[1:100,1])
    column.name <- paste('Lof_k=', i*5-5)
    names(Top100LofNode)[ncol(Top100LofNode)] <- column.name
}

Top100LofNode <- Top100LofNode[,2:51]

# Create a matrix comparing each of the differnt LOF approaches
# (each k value) with each other (how many match)

MatchMatrixNode <- matrix(rep(0,2500), ncol = 50) # template to hold results

i = 1
```

```
j = 1
for(i in 1:50){
  for(j in 1:50){
    matchmatchNode <- sum(Top100LofNode[,i] %in% Top100LofNode[,j])
    MatchMatrixNode[i,j] <- matchmatchNode
  }
}

MatchMatrixNode <- as.data.frame(MatchMatrixNode)

write.csv(MatchMatrixNode, file = "LOFcompNode 5.csv")
```

```{r Plot Node based results of different k values, include=TRUE}
ggplot(LOF.Node.results, aes(LOF.Node.results[,2])) +
  geom_histogram() + labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFNodek5.pdf")
ggplot(LOF.Node.results, aes(LOF.Node.results$`k = 30`)) +
  geom_histogram() + labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFNodek30.pdf")
ggplot(LOF.Node.results, aes(LOF.Node.results$`k = 60`)) +
  geom_histogram() + labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFNodek60.pdf")
ggplot(LOF.Node.results, aes(LOF.Node.results$`k = 90`)) +
  geom_histogram() + labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFNodek90.pdf")
ggplot(LOF.Node.results, aes(LOF.Node.results$`k = 120`)) +
  geom_histogram() + labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFNodek120.pdf")
ggplot(LOF.Node.results, aes(LOF.Node.results$`k = 150`)) +
     geom_histogram() + labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFNodek150.pdf")
ggplot(LOF.Node.results, aes(LOF.Node.results$`k = 180`)) +
  geom_histogram() + labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFNodek180.pdf")
ggplot(LOF.Node.results, aes(LOF.Node.results$`k = 210`)) +
  geom_histogram() + labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFNodek210.pdf")
ggplot(LOF.Node.results, aes(LOF.Node.results$`k = 240`)) +
  geom_histogram() + labs(x =NULL , y = NULL)+ scale_y_sqrt()
ggsave("LOFNodek240.pdf")
```

B.6 Anomaly Section

```
---
title: "SGW Anomaly selection"
author: "Jason Salzwedel"
---
```

```{r Load data, include=TRUE}
library(ggplot2)
library(tidyverse)
library(caret)
library(MASS)
library(reshape2)
rm(list=ls())
Data <- read.csv("FinalDataSGW.CSV", header = TRUE, sep = ",")
Data.PCA.Prcomp <-
  read.csv("Data.PCA.Prcomp.csv", header = TRUE, sep = ",")
Data.PCA100 <- cbind(Data[,c(1:9)],Data.PCA.Prcomp[,1:100])
```


```{r GMM on All data together, include=TRUE}
# Calculate the mean and covariance and means of all varialbles

PreObject <- preProcess(Data.PCA.Prcomp[,1:100],method="center")
Data.PCA.Prcomp.Center <- predict(PreObject,Data.PCA.Prcomp[,1:100])
m.Data.PCA.Prcomp.Center= as.matrix(Data.PCA.Prcomp.Center)

pca <- Data.PCA.Prcomp[,1:100]
pca= as.matrix (pca)

sigma1 =cov(pca)

#calculate the probability density function and probilities of all points.
X <- (2*pi)^(-ncol(m.Data.PCA.Prcomp.Center)/2)*det(sigma1)^(-0.5)
Y <- exp(-0.5 *rowSums((m.Data.PCA.Prcomp.Center%*%
  ginv(sigma1))*Data.PCA.Prcomp.Center))
GMM <- X*Y
GMM <- as.data.frame(GMM)
X <- c(1:16070)
GMM <- cbind(X,GMM)
```


```{r GMM on All Node based, include=TRUE}
Data.PCA100 <- cbind(Data[,c(1:9)],Data.PCA.Prcomp[,1:100])

Nodes <- unique(Data.PCA100$SGW_ID)

Data.index <- data.frame(x = integer,
                         P = double)
for (Node in Nodes){
  # Calculate the mean and covariance and means of all variables
  # in this subset
    subset <- Data.PCA100 %>% filter(SGW_ID == Node)
    subset <- subset[,c(10:109)]
    PreObject <- preProcess(subset,method="center")
```

```
    Data.PCA.Prcomp.Center <- predict(PreObject,subset)
    m.Data.PCA.Prcomp.Center <- as.matrix(Data.PCA.Prcomp.Center)
    pca <- as.matrix (subset)
    sigma1<- cov(pca)

    # Calculate the probability density function and probabilities
    # of all points.
    X=(2*pi)^(-ncol(m.Data.PCA.Prcomp.Center)/2)*det(sigma1)^(-0.5)
    Y = exp(-0.5 *rowSums((m.Data.PCA.Prcomp.Center%*%
      ginv(sigma1))*Data.PCA.Prcomp.Center))
    psubset=X*Y
    index  <- Data.PCA100 %>% filter(SGW_ID == Node) %>% dplyr::select(X)
    Data.index <- rbind(Data.index, cbind (index, psubset))
}

GMM.Node <- as.data.frame(Data.index)
colnames(GMM.Node) <- c("X","GMM.Node")
```

````
```{r MGD on All Node and time based, include=TRUE}

# The following code attempts to create the node/time based calculations,
# but as there are only 13 observations at this level and that there
# are 100 variables, sigma is not invertible. This results in "inf"
# values be calculated

Data.PCA100 <- cbind(Data[,c(1:9)],Data.PCA.Prcomp[,1:100])

Nodes <- unique(Data.PCA100$SGW_ID)
Times <- unique(Data.PCA100$Time)
Data.index <- data.frame(x = integer,
                         P = double)

for (Node in Nodes[1]){
  for (TOD in Times[2]){
    subset <- Data.PCA100 %>% filter(SGW_ID == Node & Time ==TOD)
    subset <- subset[,c(10:109)]
    PreObject <- preProcess(subset,method="center")
    Data.PCA.Prcomp.Center <- predict(PreObject,subset)
    m.Data.PCA.Prcomp.Center <- as.matrix(Data.PCA.Prcomp.Center)
    pca <- as.matrix (subset)
    sigma1<- cov(pca)

    # Calculate the probability density function and probabilities
    # of all points.

    X <- (2*pi)^(-ncol(m.Data.PCA.Prcomp.Center)/2)*
      det(sigma1)^(-0.5)/10000000
    Y <- exp(-0.5 *rowSums((m.Data.PCA.Prcomp.Center%*%
````
````

```
      ginv(sigma1))*Data.PCA.Prcomp.Center))
    psubset <- X*Y
    index  <- Data.PCA100 %>% filter(SGW_ID == Node & Time ==TOD) %>%
      dplyr::select(X)
    Data.index <- rbind(Data.index, cbind (index, psubset))
  }
}
    write.csv(Data.index, file = "GMM Node time.csv")
```

```{r add GMM results to master data set, include=TRUE}

updatedmasteroutlierresults <-
  inner_join(updatedmasteroutlierresults,GMM, by = "X")
updatedmasteroutlierresults <-
  inner_join(Master_Outlier_results,GMM.Node, by = "X")
    write.csv(updatedmasteroutlierresults,
      file = "Final_Master_Outliers.csv")
```

```{r check for comonanlity between all ten approaches, include=TRUE}
library(tidyverse)
Outliers.Data <- read.csv(file = "Final_Master_Outliers.csv")
Outliers.Data <- select(Outliers.Data, -"X.1")

# The OCSVM and GMM based approaches rank outliers with negative
# inversely. To align with the other approaches, the numbers will
# be multiplied by -1


Outliers.Data$OCSVM.nu0.13.gamma.0.002 <-
-X-1*Outliers.Data$OCSVM.nu0.13.gamma.0.002
Outliers.Data$OCSVM.Node.nu0.5.gamma.1e.04 <-
-1*Outliers.Data$OCSVM.Node.nu0.5.gamma.1e.04
Outliers.Data$OCSVM.Node.Time.nu0.5.gamma.1e.04 <-
-1*Outliers.Data$OCSVM.Node.Time.nu0.5.gamma.1e.04
Outliers.Data$GMM <- -1*Outliers.Data$GMM
Outliers.Data$GMM.Node <- -1*Outliers.Data$GMM.Node
Outliers.Data <- Outliers.Data %>% dplyr::select(-X.1)

MatchMatrixNode <- matrix(rep(0,100), ncol = 10) #  To hold results

for(i in 2:11){
  for(j in 2:11){
    matchmatchNode <- sum(arrange(Outliers.Data,
      desc(Outliers.Data[,i]))[1:100,1] %in%
        arrange(Outliers.Data, desc(Outliers.Data[,j]))[1:100,1])
    MatchMatrixNode[i-1,j-1] <- matchmatchNode
  }
}
```

```
```

B.7 Anomaly Selection

```
---
title: "SGW Anomaly Selection"
author: "Jason Salzwedel"
---

# This chapter of code starts with investigating the results of the
# various anomaly detection techniques. This is with the aim of
# select a final list of outlier observations to be removed from
# the original data set. This results in the creation of a
# anomaly free data set to be in the next semi supervised
# section of the project

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
rm(list=ls())

# Read in the results obtained for the outlier detection approaches
# completed
Outliers.Data <-
  read.csv("Final_Master_Outliers.CSV", header = TRUE, sep = ",")
```


#The results from the various approaches are arranged according
#to ranking and then plotted separately. This given a view to
#investigate how many observations can be considered to be outliers

```{r Plot the sorted results, include=FALSE}
library(ggplot2)
library(tidyverse)

# The OCSVM based approaches rank outliers with negative numbers.
# To allign with the other approaches, the numbers will be multiplied
# by -1

Outliers.Data$OCSVM.nu0.13.gamma.0.002 <-
  -1*Outliers.Data$OCSVM.nu0.13.gamma.0.002
Outliers.Data$OCSVM.Node.nu0.5.gamma.1e.04 <-
  -1*Outliers.Data$OCSVM.Node.nu0.5.gamma.1e.04
Outliers.Data$OCSVM.Node.Time.nu0.5.gamma.1e.04 <-
  -1*Outliers.Data$OCSVM.Node.Time.nu0.5.gamma.1e.04
Outliers.Data$GMM <- -1*Outliers.Data$GMM
Outliers.Data$GMM.Node <- -1*Outliers.Data$GMM.Node
Outliers.Data <- Outliers.Data %>% dplyr::select(-X.1)

# Add anomaly rank for each approach
```
```

```
Anomaly.Approaches <- colnames(Outliers.Data[-1])
for (i in 1:length(Anomaly.Approaches)){
  Rank.col.name <- paste0('Rank',Anomaly.Approaches[i])
  print(Rank.col.name)
  Outliers.Data <- Outliers.Data %>%
    mutate(Rank = dense_rank(desc(Outliers.Data[i+1])))
  colnames(Outliers.Data)[length(Outliers.Data)] <- Rank.col.name
}

# Plot approachs
for (i in 2:((length(Outliers.Data-1)/2))+1){
  plot.name <- paste0(colnames(Outliers.Data[i]),'.pdf')
  ggplot(Outliers.Data, aes(y = Outliers.Data[,i],
    x = Outliers.Data[,i+length(Outliers.Data-1)/2])) +
    geom_line() +
    labs(x =NULL , y = NULL)
  ggsave(file = plot.name)
}

# Based on plots 150 points will be removed as outliers
```

```{r Select Outliers, include=FALSE}
# Add Sum of rankings to data set.
Sum.Of.Ranks <- Outliers.Data %>%
  dplyr::select(starts_with('Rank')) %>%
    mutate(Sum.Of.Ranks = rowSums(.)) %>%
      dplyr::select (Sum.Of.Ranks)
 Outliers.Data <- cbind(Outliers.Data,Sum.Of.Ranks)

Outlier.Data.150 <- Outliers.Data %>% arrange(Sum.Of.Ranks)
Outlier.Data.150 <- Outlier.Data.150[1:150,]
Outlier.Data.150.rank <- Outlier.Data.150 %>%
  dplyr::select(starts_with('Rank'))

#Compare each approach top 150 with the 150 outliers selected.
for (i in 1:length(Outlier.Data.150.rank)){

  print (sum(as.matrix(Outlier.Data.150.rank[i]) %in% c(1:150)))
}

write.csv(Outlier.Data.150, file = "final150outliers.csv")
```

```
#The following code identifies the outliers in the clean data set.
#The Element_id and the time stamp are then extracted and used to
#remove the anomalous observations from the original raw data set.
```

```{r Remove outliers from Original dataset, include=FALSE}
```

```
#Sort data based on Sum.of.rank and extract the observation numbers
Outlier.Data.sorted <- Outliers.Data %>% arrange(Sum.Of.Ranks)
Outlier.Observations <- as.data.frame(Outlier.Data.sorted$X[1:150])
colnames(Outlier.Observations) <- "X"

#join these observation numbers "X" on the orignial cleaned data,
# leaving only the outliers.
Original.Clean.Data <-
  read.csv("FinalDataSGW.CSV", header = TRUE, sep = ",")
Anomalies.in.Original.Clean.Data <-
  inner_join(Original.Clean.Data, Outlier.Observations, by ="X" )

# read in the Original raw data returned by the SQL query
Raw.Data <-
  read.csv("../SQL/SQLResults/SGW14days.csv", header = TRUE, sep = ",")

Raw.Data.Less.Outliers <-
  anti_join(Raw.Data, Anomalies.in.Original.Clean.Data,
  by = c("SGW_ID","STARTTIME"))

#write.csv(Raw.Data.Less.Outliers, file = "RawDataLessOutliers.csv")
```

B.8 Autoencoder Creation for Anomaly Detection

```
---
title: "SGW Autoencoder"
author: "Jason Salzwedel"
---

```{r Update h2O, include=TRUE}
# This section copied from the H2o support page removes previously
# installed H2o Packages, then download and installs the latest H2o
# packages.
# The following two commands remove any previously installed H2O
# packages for R.
if ("package:h2o" %in% search()) { detach("package:h2o", unload=TRUE) }
if ("h2o" %in% rownames(installed.packages())) { remove.packages("h2o") }

# Next, we download packages that H2O depends on.
pkgs <- c("RCurl","jsonlite")
for (pkg in pkgs) {
if (! (pkg %in% rownames(installed.packages()))) { install.packages(pkg) }
}

# Now we download, install and initialize the H2O package for R.
install.packages("h2o", type="source",
  repos="http://h2o-release.s3.amazonaws.com/h2o/rel-wolpert/11/R")
```
```

```
# Finally, let's load H2O and start up an H2O cluster
library(h2o)
#h2o.init()
```

```{r Function definitions, include=TRUE}

# These functions receive two sets of observations and their associated
# anomaly values. Observations are sorted based on Anomaly levels and
# the top 100 are compared. The number of common top 100 observations
# between these two sets are returned.

# This function sorts the first set in ascending order (for anomalies
# that inversely proportional to the anomaly size)

common.top.x <- function(top.x, set1, set2){
  sorted.set1 <- set1 %>% arrange ((set1[,2]))
  sorted.set2 <- set2 %>% arrange (desc(set2[,2]))
  sorted.set1 <- sorted.set1[1:top.x,]
  sorted.set2 <- sorted.set2[1:top.x,]
  in.both <- sum(sorted.set1[,1] %in% sorted.set2[,1])
  return(in.both)
}

# This function sorts the first set in descending order
# (for anomalies that proportional to the anomaly size)

common.top.x.desc <- function(top.x, set1, set2){
  sorted.set1 <- set1 %>% arrange (desc(set1[,2]))
  sorted.set2 <- set2 %>% arrange (desc(set2[,2]))
  sorted.set1 <- sorted.set1[1:top.x,]
  sorted.set2 <- sorted.set2[1:top.x,]
  in.both <- sum(sorted.set1[,1] %in% sorted.set2[,1])
  return(in.both)
}
```



```{r Load data, include=TRUE}
library(ggplot2)
library(tidyverse)
rm(list=ls())
Clean.Original.Data <-
  read.csv("RawDataLessOutliers.csv", header = TRUE, sep = ",")
Clean.Original.Data <- Clean.Original.Data %>% select( -X)
```

```{r Feature enigeering , echo=FALSE}

# Add features to the data set to modify contextual anomaly
```

```
# detection to point anomaly detection
library(lubridate)
library(plotly)
Clean.Original.Data.Features <- Clean.Original.Data %>%
  mutate(Date.Time =
    as.POSIXct(as.character(levels(Clean.Original.Data$STARTTIME)),
    format = "%Y-%m-%d %H:%M")[Clean.Original.Data$STARTTIME]) %>%
  mutate(Day = wday(Date.Time))%>%
  mutate(Hour = lubridate::hour(Date.Time))%>%
  mutate(Qtr = minute(Date.Time))%>%
  mutate(Qtr = replace(Qtr, Qtr==0,"00"))%>%
  mutate(Time = paste(Hour,Qtr, sep = ":"))%>%
  mutate(Day = as.factor(Day))%>%
  mutate(Hour = as.factor(Hour))%>%
  mutate(Qtr = as.factor(Qtr))%>%
  mutate(Time = as.factor(Time))%>%
  select(Time,Qtr,Hour,Day,Date.Time,  everything()) %>%
  select(-Date.Time)
```

```{r Initialise H2o and train test data, echo=F, eval=F}
# NN analysis done of Raw data set

## Load all packages first
library(h2o)

library(reshape2)

## Initialise H2O Connection
## Start a local H2O cluster directly from R
localH2O = h2o.init(ip = "localhost", port = 54321,
  startH2O = TRUE,min_mem_size = "3g",nthreads = 5)


# The Variable "Day" represents the day of the week and
# must be changed to factor.

# Convert the data frame into an h2O object
Clean.Original.Data.Features.h2o <- as.h2o(Clean.Original.Data.Features)

# Split the data into Train, validation and test data sets
Clean.Original.Data.Features.h2o.split <-
  h2o.splitFrame(Clean.Original.Data.Features.h2o, ratios = c(0.1,0.1))
Test.Clean.Original.Data.Features.h2o <-
  Clean.Original.Data.Features.h2o.split[[1]]
Validate.Clean.Original.Data.Features.h2o <-
  Clean.Original.Data.Features.h2o.split[[2]]
Train.Clean.Original.Data.Features.h2o <-
  Clean.Original.Data.Features.h2o.split[[3]]
```

```{r Create Hyperparameter Grid and train AutoEncoders, echo=F, eval=F}

# Create parameter list for Grid Search
Autoencoder.Parameter.1 <- list(

                                hidden = list(c(1158)),
                                l1 = c(0),
                                 hidden_dropout_ratios = list(c(0))
)


# Train all the models specified in the Grid parameter list
# The X values include "time","qtr","hour", "day", "region", "element_id"
s <- proc.time()
Grid.Search.1 <- h2o.grid(algorithm = "deeplearning",
              autoencoder = TRUE,
              x = c(1:4,6:1001),
              grid_id = "Grid.Search.1",
              training_frame = Train.Clean.Original.Data.Features.h2o,
              validation_frame = Validate.Clean.Original.Data.Features.h2o,
              hyper_params = Autoencoder.Parameter.1,
              activation = "TanhWithDropout",
              epochs =10,
              reproducible = FALSE,
              seed = 24
              )
proc.time() - s

#Get the Grid results
Grid.Result.search.1 <- h2o.getGrid(grid_id = "Grid.Search.1",
                                    sort_by = "mse",
                                    decreasing = TRUE)
```

```{r Compare results using line plots, echo=F}
# In chosing which model to choose, one should go beyond the MSE only.
# Two models with the same MSE could have different different distributions.
# The approach to be take is to select the model the has the lowest average
# MSE for the majority of the observations and then a high MSE for the
# remaining few.


Sorted.concat.S1 <- c(1:15920)
Sorted.concat.S1 <- as.data.frame(Sorted.concat.S1)
names(Sorted.concat.S1)[1] <- 'X'

for (i in 1:nrow(Grid.Result.search.1@summary_table)){
  Model.i <- h2o.getModel(Grid.Result.search.1@model_ids[[i]])


```

```
  AE.anomaly <-
    h2o.anomaly (Model.i,Clean.Original.Data.h2o , per_feature = FALSE)
  AE.anomaly <- as.data.frame(AE.anomaly)
  AE.anomaly <- arrange(AE.anomaly, desc(AE.anomaly$Reconstruction.MSE))
  Sorted.concat.S1 <- cbind(Sorted.concat.S1, AE.anomaly)
  names(Sorted.concat.S1)[i+1] <- paste0('L1_'
                                ,Model.i@parameters$l1
                                ,' l2_', Model.i@parameters$l2
                                ,' Hidden_'
                                ,paste(as.character(Model.i@parameters$hidden)
                                , collapse = ' ')
                                ,' HDR_'
                                ,Model.i@parameters$hidden_dropout_ratios
                                )
}

write.csv(Sorted.concat.S1, file = "GridResultsSearch_1.csv")


Plot.S <-  ggplot(data = Sorted.concat.S1[1:500,],
  mapping = aes(x = Sorted.concat.S1[1:500,1],
    y = Sorted.concat.S1[1:500,2], ymin =0.02, ymax = 0.03 ) ) +
    geom_line()

for(i in 1:(nrow(Grid.Result.search.1@summary_table )-1)){

  Plot.S <- Plot.S + geom_line(data = Sorted.concat.S1[1:500,],
    aes(x = Sorted.concat.S1[1:500,1], y = Sorted.concat.S1[1:500,i+2] ))
  }

Plot.S
```

```{r Compare results using line plots, echo=F}
# add features to the data set to modify contextual anomaly detection
# to point anomaly detection

Live.Data <- read.csv("../SQL/SQLResults/LiveSGW.CSV",
  header = TRUE, sep = ",")

library(lubridate)
library(plotly)
Live.Data.Features <- Live.Data %>%
  mutate(Date.Time =
  as.POSIXct(as.character(levels(Live.Data$STARTTIME)),
  format = "%Y-%m-%d %H:%M")[Live.Data$STARTTIME]) %>%
  mutate(Day = wday(Date.Time))%>%
  mutate(Hour = lubridate::hour(Date.Time))%>%
  mutate(Qtr = minute(Date.Time))%>%
  mutate(Qtr = replace(Qtr, Qtr==0,"00"))%>%
```

```
  mutate(Time = paste(Hour,Qtr, sep = ":"))%>%
  mutate(Day = as.factor(Day))%>%
  mutate(Hour = as.factor(Hour))%>%
  mutate(Qtr = as.factor(Qtr))%>%
  mutate(Time = as.factor(Time))%>%
  select(Time,Qtr,Hour,Day,Date.Time,  everything()) %>%
  select(-Date.Time)

Live.Data.Features.h2o <- as.h2o(Live.Data.Features)

Model.1 <- h2o.getModel(Grid.Result.search.1@model_ids[[1]])
AE.anomaly <-
  h2o.anomaly (Model.1,Live.Data.Features.h2o , per_feature = FALSE)

AE.anomaly.DF <- as.data.frame(AE.anomaly)
AE.anomaly.DF.node.time <- cbind (AE.anomaly.DF,Live.Data.Features)

write.csv(AE.anomaly.DF.node.time, file = "LiveAnomalies.csv")

AE.Variable.importance <- h2o.varimp(Model.1)

write.csv(AE.Variable.importance, file = "Variable importance.csv")

h2o.mse(Model.1)
h2o.rmse(Model.1)
```

```{r Compare Modified SGW LTE Results, echo=F}
# The Data is modified by copying the values of the G19M10C6 of
# 10403536 seen at 2018-07-02 12:00 on CTN GGCT04 to the interval
# at 2018-06-25 12:00 on CTN  GGCT04 which was zero.
# This is to test if this comes up as an anomaly.

# Add features to the data set to modify contextual anomaly detection
# to point anomaly detection

Live.Data <-
  read.csv("SGW14daysPostG19M10C6Mod.csv", header = TRUE, sep = ",")

library(lubridate)
library(plotly)
Live.Data.Features <- Live.Data %>%
  mutate(Date.Time =
    as.POSIXct(as.character(levels(Live.Data$STARTTIME)),
    format = "%Y-%m-%d %H:%M")[Live.Data$STARTTIME]) %>%
  mutate(Day = wday(Date.Time))%>%
  mutate(Hour = lubridate::hour(Date.Time))%>%
  mutate(Qtr = minute(Date.Time))%>%
  mutate(Qtr = replace(Qtr, Qtr==0,"00"))%>%
```

```
  mutate(Time = paste(Hour,Qtr, sep = ":"))%>%
  mutate(Day = as.factor(Day))%>%
  mutate(Hour = as.factor(Hour))%>%
  mutate(Qtr = as.factor(Qtr))%>%
  mutate(Time = as.factor(Time))%>%
  select(Time,Qtr,Hour,Day,Date.Time,  everything()) %>%
  select(-Date.Time)

Live.Data.Features.h2o <- as.h2o(Live.Data.Features)


Model.1 <- h2o.getModel(Grid.Result.search.1@model_ids[[1]])
AE.anomaly <-
  h2o.anomaly (Model.1,Live.Data.Features.h2o , per_feature = FALSE)
Predictions <- h2o.predict(Model.1, newdata = Live.Data.Features.h2o)
Predictions.DF <- as.data.frame(Predictions)
Predictions.DF.Original <-
  cbind(AE.anomaly.DF,Live.Data.Features, Predictions.DF)

AE.anomaly.DF <- as.data.frame(AE.anomaly)
AE.anomaly.DF.node.time <- cbind (AE.anomaly.DF,Live.Data.Features)

write.csv(AE.anomaly.DF.node.time, file = "LiveAnomalies.csv")

AE.Variable.importance <- h2o.varimp(Model.1)

write.csv(AE.Variable.importance, file = "Variable importance.csv")
write.csv(Predictions.DF.Original, file = "h2oPredictions.csv")
h2o.mse(Model.1)
h2o.rmse(Model.1)
```

B.9 Analysis of Results

```
---
title: "SGW Analysis of Results"
author: "Jason Salzwedel"
---


```{r Load data, include=TRUE}
library(ggplot2)
library(tidyverse)
library(plotly)
library(scales)
rm(list=ls())
 AE.anomaly.DF.node.time <-
  read.csv("LiveAnomaliesv2.csv", header = TRUE, sep = ",")
 AE.anomaly.DF.node.time <-  AE.anomaly.DF.node.time %>% select( -X)
```
```

```{r Plot MSE values, echo=F}

# To plot a trend over time a Variable of type POSIXct is required.
AE.anomaly.DF.node.time.POSIXct <- AE.anomaly.DF.node.time %>%
  mutate(Date.Time =
    as.POSIXct(as.character(levels(AE.anomaly.DF.node.time$STARTTIME)),
    format = "%Y-%m-%d %H:%M")[AE.anomaly.DF.node.time$STARTTIME])

# the first set of observations to be analysed are those with an MSE above 1
above1 <- AE.anomaly.DF.node.time.POSIXct %>%
  dplyr::filter(Reconstruction.MSE > 1)
SGW <- ggplot(above1,
  aes(x = Date.Time, y= Reconstruction.MSE, color = SGW_ID)) +
  geom_point()+ ylim (0, 100) + xlim(c(as.POSIXct('2018-06-22 09:00:00',
  format = "%Y-%m-%d %H:%M:%S"),
  as.POSIXct('2018-07-06 08:30:00', format = "%Y-%m-%d %H:%M:%S")))
SGW
ggsave("above1.pdf")
ggplotly(SGW)

# A subset of these are for the daily spikes at 9am on the 22nd
Promo <- AE.anomaly.DF.node.time.POSIXct %>%
  dplyr::filter(Reconstruction.MSE > 1 &
  Date.Time >= as.Date("2018-6-22")  &
  Date.Time <= as.Date("2018-6-23"))
SGW <- ggplot(Promo,
  aes(x = Date.Time, y= Reconstruction.MSE, color = SGW_ID)) +
  geom_point()+ ylim (0, 100) + xlim(c(as.POSIXct('2018-06-22 09:00:00',
  format = "%Y-%m-%d %H:%M:%S"),
  as.POSIXct('2018-07-06 08:30:00', format = "%Y-%m-%d %H:%M:%S")))
SGW
ggsave("Promo.pdf")

# To investigate why these points are outliers they are compared to
# normal intervals the following Thursday.

Promo <- AE.anomaly.DF.node.time.POSIXct %>%
  dplyr::filter(Reconstruction.MSE > 1 &
    Date.Time >= as.Date("2018-6-22")  &
    Date.Time <= as.Date("2018-6-23") & Hour >= "13" &
    Hour <= "17" & SGW_ID %in% c("GGCT03","GGCT04","GGDN03","GGDN04",
    "GGMT03","GGMT04","GGPS03","GGPS04","NFV1-GGMD01","NFV1-GGPR02"))

PromoRef <- AE.anomaly.DF.node.time.POSIXct %>%
  dplyr::filter(Date.Time >= as.Date("2018-6-28")  &
  Date.Time <= as.Date("2018-6-29") & Hour >= "13" &
  Hour <= "17" & SGW_ID %in% c("GGCT03","GGCT04","GGDN03","GGDN04","GGMT03",
```

```
  "GGMT04","GGPS03","GGPS04","NFV1-GGMD01","NFV1-GGPR02"))

# Caluculate the Average of each type and compare them through a ratio
PromoAVG <- Promo %>%
  select(-c(Time,Day,STARTTIME,Reconstruction.MSE,Date.Time,Hour,Qtr,
  REGION, SGW_ID))
PromoAVG <- colMeans(PromoAVG)

PromoRefAVG <- PromoRef %>%
  select(-c(Time,Day,STARTTIME,Reconstruction.MSE,Date.Time,Hour,Qtr,
  REGION, SGW_ID))
PromoRefAVG <- colMeans(PromoRefAVG)

PromoVSRef <- cbind (PromoAVG,PromoRefAVG )
PromoVSRef <- as.data.frame(PromoVSRef)
PromoVSRef <- PromoVSRef %>% mutate(Counter = rownames(PromoVSRef))
PromoVSRef <- PromoVSRef %>% mutate(Ratio = PromoAVG/PromoRefAVG)

# A subset of these are for the daily spikes at 9am on the Cape Town SGWs
nineAM <- AE.anomaly.DF.node.time.POSIXct %>%
  dplyr::filter(Time == "9:00") %>%
  dplyr::filter(Reconstruction.MSE > 0.16)

SGW <- ggplot(nineAM ,
  aes(x = Date.Time, y= Reconstruction.MSE, color = SGW_ID)) +
  geom_point() + ylim (0, 100)
SGW

#ggsave("nineAM.pdf")

#To check what the difference is, this observation is compared to the 9:15
# intervals in the Cape Town region

nine15AM <- AE.anomaly.DF.node.time.POSIXct %>%
  dplyr::filter(Time == "9:15") %>%
  dplyr::filter(Reconstruction.MSE < 0.16) %>%
  dplyr::filter(Reconstruction.MSE > 0.002) %>%
  dplyr::filter(REGION == "CTN")

# To investigate why these are outliers they are compared to normal
# observations.
# The averages of all the variables values are caluculated for the 9:00
# and the 9:15 intervals

nineAMAVG <- nineAM %>%
  select(-c(Time,Day,STARTTIME,
  Reconstruction.MSE,Date.Time,Hour,Qtr, REGION, SGW_ID))
nineAMAVG <- colMeans(nineAMAVG)

nine15AMAVG <- nine15AM %>%
```

```
  select(-c(Time,Day,STARTTIME,Reconstruction.MSE,Date.Time,Hour,Qtr,
  REGION, SGW_ID))
nine15AMAVG <- colMeans(nine15AMAVG)

nine.vs.nine15 <- cbind (nineAMAVG,nine15AMAVG)
nine.vs.nine15 <- as.data.frame(nine.vs.nine15)
nine.vs.nine15 <- nine.vs.nine15 %>%
  mutate(Counter = rownames(nine.vs.nine15))
nine.vs.nine15 <- nine.vs.nine15 %>%
  mutate(Ratio = nineAMAVG/nine15AMAVG)


#The next cluster are those on GGCT04 on the 25th at 5pm
GGCT04 <- AE.anomaly.DF.node.time.POSIXct %>%
  dplyr::filter(SGW_ID == "GGCT04") %>%
  dplyr::filter(Reconstruction.MSE > 2.6) %>%
  filter(Date.Time >= as.Date("2018-6-25")  &
  Date.Time <= as.Date("2018-6-26"))

lims <- as.POSIXct(strptime(c("2018-06-22 09:00", "2019-07-06 08:30"),
format = "%Y-%m-%d %H:%m"))
str(timespan)
SGW <- ggplot(GGCT04 ,
aes(x = Date.Time, y= Reconstruction.MSE, color = SGW_ID)) +
  geom_point() + ylim (0, 100) + xlim(c(as.POSIXct('2018-06-22 09:00:00',
  format = "%Y-%m-%d %H:%M:%S"),
     as.POSIXct('2018-07-06 08:30:00', format = "%Y-%m-%d %H:%M:%S")))
SGW

ggsave("GGCT04.pdf")

# To investigate why these are outliers they are compared to normal
# observations
GGCT04 <- AE.anomaly.DF.node.time.POSIXct %>%
  dplyr::filter(Hour == "17" & Qtr !="0") %>%
  dplyr::filter(Reconstruction.MSE > 2.6) %>%
  dplyr::filter(SGW_ID == "GGCT04")
  GGCT04AVG <- GGCT04 %>%
  select(-c(Time,Day,STARTTIME,Reconstruction.MSE,Date.Time,Hour,Qtr,
  REGION, SGW_ID))

GGCT04AVG <- colMeans(GGCT04AVG)

GGCT04Norm <- AE.anomaly.DF.node.time.POSIXct %>%
  dplyr::filter(Hour == "16" & Qtr !="0")  %>%
  dplyr::filter(SGW_ID == "GGCT04")
GGCT04AVGNorm <- GGCT04Norm %>%
select(-c(Time,Day,STARTTIME,Reconstruction.MSE,Date.Time,Hour,Qtr,
  REGION, SGW_ID))
GGCT04AVGNorm <- colMeans(GGCT04AVGNorm)
```

```
GGCT04OutvsNorm <- cbind (GGCT04AVGNorm,GGCT04AVG)
GGCT04OutvsNorm <- as.data.frame(GGCT04OutvsNorm)
GGCT04OutvsNorm <- GGCT04OutvsNorm %>%
  mutate(Counter = rownames(GGCT04OutvsNorm))
GGCT04OutvsNorm <- GGCT04OutvsNorm %>%
  mutate(Ratio = GGCT04AVG/GGCT04AVGNorm)

#The next cluster are those on GGCT03 on the 28th at 8pm
GGCT03 <- AE.anomaly.DF.node.time.POSIXct %>%
dplyr::filter(SGW_ID == "GGCT03") %>%
  dplyr::filter(Reconstruction.MSE > 1) %>%
  filter(Date.Time >= as.Date("2018-6-28")  &
  Date.Time <= as.Date("2018-6-29"))

lims <- as.POSIXct(strptime(c("2018-06-22 09:00", "2019-07-06 08:30"),
format = "%Y-%m-%d %H:%m"))
str(timespan)
SGW <- ggplot(GGCT03 ,
aes(x = Date.Time, y= Reconstruction.MSE, color = SGW_ID)) +
  geom_point() + ylim (0, 100) + xlim(c(as.POSIXct('2018-06-22 09:00:00',
  format = "%Y-%m-%d %H:%M:%S"),
     as.POSIXct('2018-07-06 08:30:00', format = "%Y-%m-%d %H:%M:%S")))
SGW

ggsave("GGCT03.pdf")

#To investigate why these are outliers they are compared to normal
# observerations
GGCT04 <- AE.anomaly.DF.node.time.POSIXct %>%
  dplyr::filter(Hour == "17" & Qtr !="0") %>%
  dplyr::filter(Reconstruction.MSE > 2.6) %>%
  dplyr::filter(SGW_ID == "GGCT04")
GGCT04AVG <- GGCT04 %>%
  select(-c(Time,Day,STARTTIME,Reconstruction.MSE,Date.Time,Hour,Qtr,
  REGION, SGW_ID))
GGCT04AVG <- colMeans(GGCT04AVG)

GGCT04Norm <- AE.anomaly.DF.node.time.POSIXct %>%
  dplyr::filter(Hour == "16" & Qtr !="0")  %>%
  dplyr::filter(SGW_ID == "GGCT04")
GGCT04AVGNorm <- GGCT04Norm %>%
select(-c(Time,Day,STARTTIME,Reconstruction.MSE,Date.Time,Hour,Qtr,
  REGION, SGW_ID))
GGCT04AVGNorm <- colMeans(GGCT04AVGNorm)

GGCT04OutvsNorm <- cbind (GGCT04AVGNorm,GGCT04AVG)
GGCT04OutvsNorm <- as.data.frame(GGCT04OutvsNorm)
GGCT04OutvsNorm <- GGCT04OutvsNorm %>%
  mutate(Counter = rownames(GGCT04OutvsNorm))
```

```
GGCT04OutvsNorm <- GGCT04OutvsNorm %>%
  mutate(Ratio = GGCT04AVG/GGCT04AVGNorm)

#The next cluster are those on PTA nodes on the 29th at 8pm
PTA <- AE.anomaly.DF.node.time.POSIXct %>%
  dplyr::filter(REGION == "PTA") %>%
  dplyr::filter(Reconstruction.MSE > 1) %>%
  filter(Date.Time >= as.Date("2018-6-29")  &
  Date.Time <= as.Date("2018-6-30"))

lims <- as.POSIXct(strptime(c("2018-06-22 09:00", "2019-07-06 08:30"),
  format = "%Y-%m-%d %H:%m"))

SGW <- ggplot(PTA ,
  aes(x = Date.Time, y= Reconstruction.MSE, color = SGW_ID)) +
  geom_point() + ylim (0, 100) + xlim(c(as.POSIXct('2018-06-22 09:00:00',
  format = "%Y-%m-%d %H:%M:%S"),
     as.POSIXct('2018-07-06 08:30:00', format = "%Y-%m-%d %H:%M:%S")))
ggplotly(SGW)

ggsave("PTA.pdf")


#To investigate why these are outliers they are compared to normal
# observation
Promo2 <- AE.anomaly.DF.node.time.POSIXct %>%
  dplyr::filter(Hour == "23" & Qtr =="15") %>%
  dplyr::filter(Reconstruction.MSE > 90) %>%
  dplyr::filter(SGW_ID == "NFV1-GGPR02")
Promo2Ref <- AE.anomaly.DF.node.time.POSIXct %>%
  dplyr::filter(Hour == "23" & Qtr =="0")  %>%
  dplyr::filter(SGW_ID == "NFV1-GGPR02") %>%
  filter(Date.Time >= as.Date("2018-6-29")  &
  Date.Time <= as.Date("2018-6-30"))

Promo2<- Promo2 %>%
  select(-c(Time,Day,STARTTIME,
  Reconstruction.MSE,Date.Time,Hour,Qtr, REGION, SGW_ID))
Promo2Ref <- Promo2Ref %>%
  select(-c(Time,Day,STARTTIME,Reconstruction.MSE,Date.Time,Hour,
  , REGION, SGW_ID))

Promo2 <- colMeans(Promo2)
Promo2Ref <- colMeans(Promo2Ref)

Promo2VSRef <- cbind (Promo2,Promo2Ref)
Promo2VSRef <- as.data.frame(Promo2VSRef)
Promo2VSRef <- Promo2VSRef %>% mutate(Counter = rownames(Promo2VSRef))
Promo2VSRef <- Promo2VSRef %>% mutate(Ratio = Promo2/Promo2Ref)
```

```
write.csv(Promo2VSRef, file = "promo2.csv")


# The next major group of anomalies discussed are those between 0.002 and 1
# the first set of observations to be analysed are those with an MSE
# below 0.002
above1 <- AE.anomaly.DF.node.time.POSIXct %>%
  dplyr::filter(Reconstruction.MSE < 0.002 & SGW_ID %in%
  c('GGCF02', 'GGCT03','GGCT04','GGDM02','GGDDN02','GGDN03','GGMT03','GGMT04',
  'GGPS03','GGPS04','NFV1-GGMD01','NFV1-GGPR02'))
SGW <- ggplot(above1,
  aes(x = Date.Time, y= Reconstruction.MSE, color = SGW_ID)) +
  geom_point()+ ylim (0, 0.0025) + xlim(c(as.POSIXct('2018-06-22 09:00:00',
  format = "%Y-%m-%d %H:%M:%S"),
     as.POSIXct('2018-07-06 08:30:00', format = "%Y-%m-%d %H:%M:%S")))
SGW
ggsave("Below00022.pdf")
ggplotly(SGW)

# The next major group of anomalies discussed are those between 0.002 and 1
# the first set of observations to be analysed are those with an MSE below
# 0.002

#,'GGDN03','GGDM02'

above1 <- AE.anomaly.DF.node.time.POSIXct %>%
  dplyr::filter(Reconstruction.MSE < 0.2 & SGW_ID %in% c('GGCF02', 'GGCT03','GGCT04',
  'NFV1-GGMD01', 'NFV1-GGPR02'))
SGW <- ggplot(above1,
  aes(x = Date.Time, y= Reconstruction.MSE, color = SGW_ID)) +
  geom_line()+ ylim (0, 0.2) + xlim(c(as.POSIXct('2018-06-22 09:00:00',
  format = "%Y-%m-%d %H:%M:%S"),
     as.POSIXct('2018-07-06 08:30:00', format = "%Y-%m-%d %H:%M:%S")))
SGW
ggsave("VoLTE.pdf")
ggplotly(SGW)


above1 <- AE.anomaly.DF.node.time.POSIXct %>%
  dplyr::filter(Reconstruction.MSE < 0.2 & SGW_ID %in% c('GGCF02','GGDN03','GGDM02','I
SGW <- ggplot(above1,
  aes(x = Date.Time, y= Reconstruction.MSE, color = SGW_ID)) +
  geom_line()+ ylim (0, 0.2) + xlim(c(as.POSIXct('2018-06-22 09:00:00',
    format = "%Y-%m-%d %H:%M:%S"),
     as.POSIXct('2018-07-06 08:30:00', format = "%Y-%m-%d %H:%M:%S")))
SGW
ggsave("NewSGW.pdf")
ggplotly(SGW)
```

```
above1 <- AE.anomaly.DF.node.time.POSIXct %>%
  dplyr::filter(Reconstruction.MSE < 0.2 & SGW_ID %in% c('GGDN03','GGDM02'))
SGW <- ggplot(above1,
  aes(x = Date.Time, y= Reconstruction.MSE, color = SGW_ID)) +
  geom_line()+ ylim (0, 0.2) + xlim(c(as.POSIXct('2018-06-22 09:00:00',
  format = "%Y-%m-%d %H:%M:%S"),
     as.POSIXct('2018-07-06 08:30:00', format = "%Y-%m-%d %H:%M:%S")))
SGW
ggsave("DM02DN03.pdf")
ggplotly(SGW)
```

```{r Check correlation of LTE counters, echo=F}

# Due to the low impact that the cessation of VoLTE services had on on the
# reconstruction MSE, the correlation between the VoLTE counters was
# investigated
rm(list=ls())
Original.Raw <- read.csv("SGW14Days.csv", header = TRUE, sep = ",")

Original.Raw.VoLTE <- Original.Raw %>% select(G19M10C8, G19M13C12, G19M3C4,
  G19M5C42, G19M4C9, G19M12C27, G19M2C15, G19M5C2, G19M9C26, G19M10C26,
  G19M13C30, G19M3C22, G19M10C25, G19M13C29, G19M3C21, G19M10C7, G19M13C11,
  G19M3C3,   G19M5C41, G19M12C26, G19M2C14, G19M5C1, G19M9C25, G19M10C5,
  G19M13C9, G19M3C1, G19M5C39, G19M10C6, G19M13C10, G19M3C2, G19M5C40,
  G19M12C24, G19M2C12, G19M4C54, G19M9C23, G19M12C25, G19M2C13, G19M4C55,
  G19M9C24, G19M1C39, G19M1C60, G19M16C22, G19M2C3, G19M1C50
)
```

Bibliography

[1] Frank E. Grubbs. Procedures for detecting outlying observations in samples. *Technometrics*, 11(1):1–21, 1969.

[2] Douglas M. Hawkins. *Identification of outliers / D.M. Hawkins.* Chapman and Hall London ; New York, 1980.

[3] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM Comput. Surv.*, 41(3):15:1–15:58, July 2009.

[4] Markus Goldstein and Seiichi Uchida. A comparative evaluation of unsupervised anomaly detection algorithms for multivariate data. *PLOS ONE*, 11(4):1–31, 04 2016.

[5] Fabrizio Angiulli and Clara Pizzuti. Fast outlier detection in high dimensional spaces. In Tapio Elomaa, Heikki Mannila, and Hannu Toivonen, editors, *Principles of Data Mining and Knowledge Discovery*, pages 15–27, Berlin, Heidelberg, 2002. Springer Berlin Heidelberg.

[6] Edwin M Knorr and Raymond T Ng. Algorithms for mining distance based outliers in large datasets. Citeseer, 1998.

[7] Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. Lof: Identifying density-based local outliers. *SIGMOD Rec.*, 29(2):93–104, May 2000.

[8] The mobile broadband standard.

[9] Vodacom annual results.

[10] 3GPP. Technical Specification Group Services and System Aspects; Network architecture. Technical Specification (TS) 23.002, 3rd Generation Partnership Project (3GPP), 03 2018. Version 15.0.0.

[11] S. Novczki. An improved anomaly detection and diagnosis framework for mobile network operators. In *2013 9th International Conference on the Design of Reliable Communication Networks (DRCN)*, pages 234–241, March 2013.

[12] P. Szilagyi and S. Novczki. An automatic detectin and diagnosis framework for mobile communication systems. In *IEE Transactions on Network and Service Management*, volume 9, pages 184–197, Jun 2012.

[13] P. Casas and J. Vanerio. Super learning for anomaly detection in cellular networks. In *2017 IEEE 13th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pages 1–8, Oct 2017.

[14] Pedro Casas, Pierdomenico Fiadino, and Alessandro D'Alconzo. Machine-learning based approaches for anomaly detection and classification in cellular networks. In *TMA*, 2016.

[15] M. S. Parwez, D. B. Rawat, and M. Garuba. Big data analytics for user-activity analysis and user-anomaly detection in mobile wireless network. *IEEE Transactions on Industrial Informatics*, 13(4):2058–2065, Aug 2017.

[16] B. Gajic, S. Novczki, and S. Mwanje. An improved anomaly detection in mobile networks by using incremental time-aware clustering. In *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, pages 1286–1291, May 2015.

[17] Cynthia Wagner, Jérôme François, Radu State, and Thomas Engel. Machine learning approach for ip-flow record anomaly detection. In Jordi Domingo-Pascual, Pietro Manzoni, Sergio Palazzo, Ana Pont, and Caterina Scoglio, editors, *NETWORKING 2011*, pages 28–39, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[18] D. Yang, D. Miao, X. Qin, and G. Wei. A novel anomaly detection with temporal and spatial aggregation in mobile networks. In *2016 8th International Conference on Wireless Communications Signal Processing (WCSP)*, pages 1–5, Oct 2016.

[19] Cisco. Asr 5000 system administration guide, staros release 20, 2017.

[20] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria, 2018.

[21] Karl Pearson F.R.S. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

[22] Sridhar Ramaswamy, Rajeev Rastogi, and Kyuseok Shim. Efficient algorithms for mining outliers from large data sets. *SIGMOD Rec.*, 29(2):427–438, May 2000.

[23] Mennatallah Amer, Markus Goldstein, and Slim Abdennadher. Enhancing one-class support vector machines for unsupervised anomaly detection. In *Proceedings of the ACM SIGKDD Workshop on Outlier Detection and Description*, ODD '13, pages 8–15, New York, NY, USA, 2013. ACM.

[24] V Vapnik and A Lerner. Pattern recognition using generalized portrait method. *Automation and Remote Control*, 24:774–780, 1963.

[25] C CORTES and V Vapnik. Support-vector networks. *Machine Learning*, 20:273–297, 1995.

[26] Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik. A training algorithm for optimal margin classifiers. In *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*, COLT '92, pages 144–152, New York, NY, USA, 1992. ACM.

[27] Bernhard Schölkopf, John C Platt, John Shawe-Taylor, Alex J Smola, and Robert C Williamson. Estimating the support of a high-dimensional distribution. *Neural computation*, 13(7):1443–1471, 2001.

[28] Robert P.W. Tax, David M.J.and Duin. Support vector data description. *Machine Learning*, 54(1):45–66, Jan 2004.

[29] A. K. Jain, Jianchang Mao, and K. M. Mohiuddin. Artificial neural networks: a tutorial. *Computer*, 29(3):31–44, Mar 1996.

[30] W.S. McCulloch and W. Pitts. A logical calculus of ideas immanent in nervous activity. *Bull. Mathematical Bio-physics*, 5:115–133, 1943.

[31] R. Rosenblatt. *Principles of Neurodynamics*. Spartan Books, 1962.

[32] Marvin Minsky and Seymour Papert. *Perceptrons: An Introduction to Computational Geometry*. MIT Press, 1969.

[33] J J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554–2558, 1982.

[34] Paul Werbos and Paul J. (Paul John. Beyond regression : new tools for prediction and analysis in the behavioral sciences. 01 1974.

[35] Rumelhart , David E, James Mcclelland, and James L. *Learning Internal Representation by Error Propagation*. 01 1986.

[36] Nathalie Japkowicz, Catherine Myers, and Mark Gluck. A novelty detection approach to classification. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence - Volume 1*, IJCAI'95, pages 518–523, San Francisco, CA, USA, 1995. Morgan Kaufmann Publishers Inc.

List of Figures

List of Tables